NR INDEKSU 353965
PL ISSN 0860-1674

12

Z MIKROKOMPUTEREM NA TY

Bajtek

MIESIĘCZNY DODATEK DO SZTANDARU MŁODYCH NR 12(48) GRUDZIEŃ 1989 CENA 1500 ZŁ



Prof. Andrzej Jakubowski: Czy stać nas na Krzemową Dolinę w Polsce?

KUBUŚ PUCHATEK

NOWA EUROPA

Był to niewątpliwie rok Europy. Przy czym, to co wydarzyło się w Anno Domini 1989 na naszym kontynencie, ma znaczenie nie tylko dla nas — Polaków, Rosjan, Niemców, Czechów, Słowaków, Węgrów, Rumunów, Bułgarów, Ukraińców, Litwinów — ale także dla całego świata. 31 grudnia skończył się bowiem nie tylko rok 1989; tego dnia skończyła się również pewna epoka w historii świata, epoka fascynująca i zarazem okrutna, która nie dała definitywnych rozstrzygnięć, co do tego, jaką drogą należy iść w przyszłość — dała nam jednak pewność, którą drogą iść nie należy.

Wiele jest symboli tego roku. Montując za lat 50 kronikę filmową, opisującą wydarzenia sprzed półwiecza, ów nieznany nam reżyser z przeszłości będzie miał z czego wybierać: Wałęsa i Kiszczak witający się na kilka minut przed rozpoczęciem obrad „okrągłego stołu"; czołgi radzieckie wracające przez most na Amu-darii po zakończeniu niesławnej, prawie 10-letniej interwencji w Afganistanie; młody człowiek w dżinsach i niebieskiej kurtce, siedzący okrakiem na szczycie muru berlińskiego i rozbijający go młotkiem na kawałki, przy radosnych okrzykach ze strony zgromadzonych po stronach muru tłumów; Vaclav Havel, przemawiający do milionowego tłumu zebranego w stolicy Czechosłowacji... Będą oczywiście, oprócz tych radosnych, również sceny okrutne, te, które przekazywała na cały świat „Wolna Telewizja Rumuńska"...

Ale być może oprócz tych historycznych wydarzeń, których wspólny mianownik sprowadza się do desperackiego i konsekwentnego likwidowania pozostałości stalinowskich w ZSRR i krajach Europy Wschodniej, ów nieznany nam dokumentalista z przyszłości znajdzie notkę z 13 grudnia, w której agencja TASS informowała z Nowego Jorku, że jedna z głównych korporacji z branży informatycznej, „Control Data", wystąpiła do rządu USA o uzyskanie zgody na sprzedaż do Związku Radzieckiego niektórych swoich wyrobów. W tym konkretnym przypadku chodzi o sześć nowoczesnych komputerów „Cyber-962", przystosowanych do pracy w systemach kontrolno-awaryjnych na obiektach energetyki jądrowej. Jest kilka powodów ku temu, aby uważać ten niepozorny wydawałoby się fakt za jedno z wydarzeń, które z perspektywy będzie miało dla przyszłości znaczenie wcale nie mniejsze, niż wybicie kilkunastu dziur w murze dzielącym Berlin.

Jedną z głównych, gospodarczych przyczyn porażki krajów Europy Wschodniej, było opóźnienie w rozwoju informatyki. Można już nawet dosyć precyzyjnie określić, kiedy ten błąd został popełniony — w pierwszej połowie lat sześćdziesiątych. W Związku Radzieckim (przypomnijmy, że było to za rządów Chruszczowa) uznano nawet wówczas cybernetykę za naukę „burżuazyjną", „wrogą" (!), a ludzi, którzy się nią zajmowali po prostu tępiono. Na rezultaty nie trzeba było długo czekać — technika radziecka, pozbawiona tych wszystkich korzyści jakie daje stosowanie układów scalonych, zaczęła z roku na rok odstawać coraz bardziej od światowej. Na uzyskanie tych samych efektów zaczęto tracić w ZSRR najpierw dwa potem trzy, cztery razy więcej materiałów i energii, niż w USA.

Jednocześnie przodujące kraje zachodnie postanowiły wykorzystać postęp naukowo-techniczny do walki ze Związkiem Radzieckim i jego sojusznikami. Powołano COCOM, specjalny komitet za pośrednictwem którego 17 krajów prowadziło skoordynowaną politykę odgrodzenia ZSRR od tych technologii, które decydują o tempie rozwoju gospodarczego. Oczywiście, uzasadniano tę decyzję przede wszystkim względami obronnymi. Ale jego istotą pozostawała blokada gospodarcza.

Sygnały świadczące o tym, że Zachód zmienia swoją politykę gospodarczą w stosunku do Europy Wschodniej — są w tej sytuacji sygnałami najważniejszymi z ważnych. Otwiera nam to bowiem nie tylko dostęp do najnowocześniejszych komputerów, do tych technologii, które decydują o wydajności pracy. W istocie rzeczy, otwiera to nam bowiem dostęp do przyszłości.

Waldemar Siwiński

ZA MIESIĄC:

- **TOMS TURBO DRIVE i operacje dyskowe w CPM PLUS**
- **ALIEN — wyprawa w Kosmos**
- **U biologów — LISY I KRÓLIKI**
- **Folia do Spectrum**
- **Muzyka komputerowa — nowy świat dźwięków**
- **Spis treści Bajtka w 1989 r.**
- **dużo programów, ciekawostki, stałe rubryki, listy**
- **Złota 10-tka Bajtka**
- **BYTE — 15-letni miliarder!**

Bajtek

„BAJTEK" — MIESIĘCZNY DODATEK DO „SZTANDARU MŁODYCH"

ADRES: 00-687 Warszawa, ul. Wspólna 61. Tel. 21-12-05 Przewodniczący Rady Redakcyjnej: Jerzy Domański — redaktor naczelny „Sztandaru Młodych".

ZESPÓŁ REDAKCYJNY:
Waldemar Siwiński — kierownik zespołu „Bajtka"
Marek Czarkowski — sekretarz redakcji „Bajtka"
Roman Poznański — kierownik działu klanów
Wanda Roszkowska — opr. graficzne

KLANY REDAGUJĄ:
Atari — Wojciech Zientara
Amstrad — Jonasz Mayer
Commodore — Klaudiusz Dybowski, Dominik Falkowski
Spectrum — Marian Przasnyski

STALE WSPÓŁPRACUJĄ:
Andrzej Pilaszek, Janusz Jarmoch, Marcin Borkowski, Łukasz Czekajewski, Waldemar Nowak, Michał Sobieszuk, Maciej Pietras, Marcin Bójko, Jacek Kunowski, Michał Karkuciński

Zdjęcia w numerze Leopold Dzikowski

Fotoskład — Tadeusz Olczak, Montaż offsetowy — Grażyna Ostaszewska, Korekta — Maria Krajewska, Zofia Wółtańska

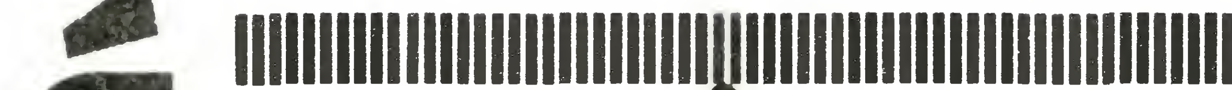
WYDAWCA: RSW „Prasa-Książka-Ruch" Młodzieżowa Agencja Wydawnicza, al. Stanów Zjednoczonych 53, 04-028 Warszawa. Telefony: Centrala 13-20-40 do 49, Redakcja Reklamy 13-20-40 do 49 w. 403, 414.

Skład techniką CRT-200, przygotowalnia offsetowa i druk: PRASOWE ZAKŁADY GRAFICZNE RSW „PRASA-KSIĄŻKA-RUCH" w Ciechanowie, ul. Sienkiewicza 51.

Nr zlecenia 96859 n. 120 000 egz.
A-111







ŚWIEŻA KREW

rozmowa z prof. dr hab. inż. Andrzejem Jakubowskim — dyrektorem Instytutu Technologii Elektronowej



Proponuję powołanie Towarzystwa Upowszechniania Komputeryzacji wśród Młodzieży

— Panie Profesorze, czy można w ogóle marzyć o uzyskaniu korzyści wiążących się z zastosowaniami informatyki — bez rozwijania u siebie własnej bazy mikroelektronicznej?

— Zdecydowanie NIE! Wprawdzie nawet poważni ludzie mówią często: skoro koszty rozwijania mikroelektroniki są tak duże, to nie rozwijajmy jej...

— Tylko kupujmy gotowe podzespoły!

— Jednak kraje, które próbowały tak postępować, wyszły na tym jak najgorzej. Na przykład, Szwajcarzy. Ponieważ produkowali znakomite zegarki mechaniczne, więc postanowili również robić zegarki elektroniczne, kupując do nich podzespoły mikroelektroniczne za granicą. Szybko się przekonali, że przestali być konkurencyjni! Okazuje się, że nie można bezkarnie zrezygnować z rozwijania tej bazy, która jest obecnie najważniejsza, a jednocześnie nie można kupić najnowocześniejszych jej wyrobów — z bazy podzespołów specjalizowanych czyli wykonywanych dla określonych, konkretnych potrzeb. Układy takie stanowią obecnie już ponad 20 proc. światowej produkcji mikroelektronicznej.

— I wszyscy w Europie tę swoją mikroelektronikę robią?

— Nie mam danych co do Albanii, ale wszystkie inne kraje mikroelektroniką się zajmują. Włącznie z Irlandią i Portugalią, które jakoś odruchowo lokujemy gdzieś na dole tabeli... Gdybyśmy chcieli wytwarzać w Polsce nowoczesny sprzęt elektroniczny, a jednocześnie nie produkować żadnych podzespołów mikroelektronicznych, to musielibyśmy zapłacić za ich import co najmniej 1 mld dolarów rocznie.

— Mówimy cały czas o układach specjalizowanych, a co z tymi, które siedzą w środku komputera?

— U nas modne były czasami takie hasła, żebyśmy ścigali świat w mikroprocesorach, czy też w pamięciach dynamicznych RAM, bo brzmi to dobrze i jest efektowną wizytówką. Wiadomo też skądinąd, że na świecie jest wielkie zapotrzebowanie na takie układy... Zapomniano jednak, że ich produkcja wymaga olbrzymich nakładów. Na świecie stać na to tylko kilka koncernów, które w dodatku zaciekle ze sobą walczą. Jest to konkurencja, w której startują tylko USA, Japonia i połączona Europa Zachodnia. Połączona, gdyż przerasta to siły pojedynczych krajów.

Nas na to nie stać. Są to bowiem najbardziej zaawansowane technologie — tzw. submikrometrowe. Musielibyśmy wybudować fabryki jakich nasz kraj jeszcze nie widział i długo jeszcze nie zobaczy (chodzi np. o superwymogi dotyczące czystości). Oczywiście, byłoby bardzo dobrze dla ogólnego rozwoju Polski, gdyby jakiś inwestor taką fabrykę chciał w Polsce postawić. Trudno przecenić, jakie miałoby to znaczenie dla podniesienia naszej kultury technicznej — są to przecież najbardziej zaawansowane technologie. Ale, nie miejmy złudzeń, dostępu do nich mieć nie będziemy — choćby dlatego, że traktowane są one póki co jako towar strategiczny.

— Co więc nam pozostaje?

— Jest cały obszar technologii 2—3 mikrometrowych (chodzi o minimalną szerokość ścieżki), które są znakomitym treningiem i przygotowaniem intelektualnym do pójścia dalej. A nie jest to wcale technologia zła, bo można w oparciu o nią produkować seryjnie np. układy pamięci 64 k.

— To, że ktoś produkuje motory, wcale nie znaczy, że mamy nie produkować rowerów?

— Oczywiście, zwłaszcza, że dobre rowery są obecnie na świecie w cenie!

— Co uzyskujemy dzięki skoncentrowaniu się na produkcji układów specjalizowanych?

— Choćby to, że dobry układ specjalizowany można sprzedać na świecie nie za 2 dolary, tak jak pamięć, tylko za 20 a nawet 100 dolarów, bo w układzie takim jest ogromny ładunek myśli konstruktora. I w tym zakresie mamy jeszcze szanse.

— Bo jest luka na świecie?

— Jest luka, ale jest również ogromna konkurencja. Układy te produkowane są przecież w wielu zakładach, o których nawet nie wiemy. Ktoś musi jednak produkować układy specjalizowane dla tych koncernów, które zajmują się produkcją samochodów czy też budową sieci telekomunikacyjnych!

— Kupując więc na przykład we Francji centrale telefoniczne, płacimy de facto również za fabrykę układów mikroelektronicznych, którą wybudowano we Francji, aby wytwarzać niezbędne podzespoły mikroelektroniczne do tych central?

— Oczywiście!

— Chce więc Pan Profesor powiedzieć, że lepiej byłoby tę fabrykę od razu zbudować u nas?

— To jest w gruncie rzeczy „być albo nie być" polskiej elektroniki. Nie można bez tego marzyć o produkcji aparatury kontrolno-pomiarowej. Obecnie, zamiast wielu poprzednio stosowanych układów, w takim urządzeniu znajduje się jeden układ scalony, nie opisany, w którym zaszyfrowana jest cała tajemnica. I jak my sobie wyobrażamy konkurencję na rynku aparatury kontrolno-pomiarowej bez możliwości wyprodukowania takich „kości"?

— I do tego wystarczy linia 3-mikrometrowa?

— Kupienie linii 3-mikrometrowej jest warunkiem uchronienia polskiej mikroelektroniki przed zagładą. Daje to bowiem szansę zachowania zespołów ludzkich i ich dorobku — a to z kolei warunkuje możliwość posunięcia się w przyszłości dalej. Przy czym nie jest tak, że my nic nie mamy. Mamy dużo elementów takiej linii. Sporo trzeba jednak jeszcze dokupić.

— Wspomniał Pan Profesor o kadrach. Gdzie one są?

— Jest jeszcze kadra w Instytucie Technologii Elektronowej. Jest też w Politechnice Warszawskiej. Ale jest to już kadra nieco zmęczona niepewnością co do swoich losów. I mocno przetrzebiona. Dlatego trzeba zrobić wszystko, żeby przyciągnąć do tej fascynującej dziedziny najzdolniejszą młodzież. Polskiej mikroelektronice potrzebna jest świeża krew! Musimy zrobić wszystko, żeby najlepsi absolwenci wyższych uczelni podejmowali pracę w tego typu instytutach, żeby najzdolniejsi absolwenci szkół średnich wybierali te kierunki studiów, które są niezbędne dla rozwoju elektroniki, żeby uczniowie najstarszych klas podstawówek też już starali się zbliżyć na serio do fizyki, matematyki, informatyki, elektroniki ze wszystkimi jej specjalnościami, chemii.

— Na ile, odbudowując polską mikroelektronikę, można liczyć na tych, którzy wyjechali i pracują za granicą?

— Na tych, których ja znam, wychowanków i pracowników Zakładu Mikroelektroniki Politechniki Warszawskiej, można liczyć na pewno. Prowadziłem już takie rozmowy z moimi młodszymi kolegami, profesorami: Andrzejem Strójwąsem, Wojciechem Małym i Jerzym Różyłło, którzy pracują w USA i odnieśli tam znaczące sukcesy. Chętnie nam pomogą. Już zresztą pomagają — przysyłają literaturę itp. Ich nazwiska są gwarancją dla międzynarodowego biznesu, że warto będzie w Polsce zainwestować. Ale, oczywiście, jeśli nie będziemy mieli dobrej kadry, tutaj, w Polsce, to nie dostaniemy żadnych zamówień ze świata. Musimy więc zrobić wszystko, żeby zapewnić sobie ten dopływ świeżej krwi.

— Ale coraz mniej osób stara się o przyjęcie na wydziały fizyki czy informatyki!

— Trzeba więc im stworzyć perspektywę pracy w tych dziedzinach. Żeby nie bali się tych kierunków studiów — i żeby wiedzieli, iż w przyszłości można na tym nieźle zarobić.

— Twórzmy tę perspektywę razem!

— Proszę bardzo! „Bajtek" ma tak dobrą markę wśród młodzieży, że połączenie naszych sił wydaje się wielce obiecujące — widzę zresztą jak mój syn skrzętnie zbiera „Bajtka", wertuje go i tłucze potem nocami na tym swoim „Commodore"... Wasi Czytelnicy wiedzą, że bez mikroelektroniki przestaniemy liczyć się w europejskiej rodzinie narodów, a to oznacza bezpowrotne utracenie szans na suwerenność, tę suwerenność prawdziwą — gospodarczą.

— Nie tylko czytajmy z wypiekami na twarzy o kalifornijskiej Krzemowej Dolinie, ale zacznijmy wreszcie tworzyć tę dolinę również w Polsce!?

— Należy ułatwiać młodzieży nie tylko kontakt z komputerami, ale również pokazywać to, co jest przed i po. Pokażmy choćby młodym ludziom jak się wytwarza układy scalone, jakich urządzeń i spełnienia jakich warunków to wymaga, związanych choćby z czystością...

— A czy Pan Dyrektor mógłby zaprosić Czytelników „Bajtka" do swojego Instytutu, żeby mogli wreszcie tę linię zobaczyć?

— Oczywiście. Chętnie pokażemy przy okazji, jak się wytwarza elementy optoelektroniki, jak wygląda system sterowania produkcją...

— Trzymamy za słowo!

— A dlaczego nie mielibyśmy iść dalej? Proponuję, aby „Bajtek" powołał Towarzystwo Upowszechniania Komputeryzacji wśród Młodzieży, które pomoże najzdolniejszym młodym ludziom zapoznać się z najnowocześniejszą techniką, w ogóle ze światem...

— Czy zechce się Pan Profesor w taką inicjatywę włączyć?

— Proszę uważać, że już się w nią włączyłem! Mam w tym zresztą swój własny interes: wierzę, że niektórzy z tych, którzy teraz czytają „Bajtka", trafią po ukończeniu studiów do naszego Instytutu i będą tą właśnie świeżą krwią, której polskiej mikroelektronice tak bardzo brakuje!

rozmawiał:
Waldemar Siwiński

„Nawet jeśli nigdy nie kupię sobie Rolls-Royce'a, to chcę wiedzieć jak on wygląda, ile cylindrów ma jego silnik i jakie są inne parametry techniczne tego pojazdu."

BATERYJNA DRUKARKA firmy Toshiba



Komputer przenośny typu laptop jest urządzeniem wystarczająco ciężkim, aby zniechęcić osobę posługującą się nim do zabierania w podróż dodatkowej, normalnej drukarki. Biorąc także pod uwagę konieczność zasilania sieciowego nie jest to na pewno rozwiązanie godne polecenia.

Co innego, jeśli kupimy drukarkę Expresswriter 301 japońskiej firmy Toshiba. Waży ona tylko dwa kilo i umożliwia pracę z ładowalnymi bateriami akumulatorów. 24-igłowa głowica pozwala na uzyskanie wydruków wysokiej jakości. Drukarka jest wyposażona w interface równoległy typu Centronics, pięć rezydentnych zestawów czcionek i pracuje w trybie draft z szybkością 60 znaków na sekundę, czyli 2—3 wolniej niż typowe urządzenia stacjonarne.



KLAWIATURA Z MYSZĄ

Korzystasz chętnie z myszy, ale jej długi przewód przeszkadza Ci w pracy. Poszukujesz mniej kłopotliwego i bardziej zwartego rozwiązania.

Za sumę $99.95 możesz nabyć klawiaturę wyposażoną dodatkowo w urządzenie myszopodobne, tzn. ruchomą kulę, której obroty wokół dowolnej osi powodują zmianę położenia kursora. Klawiatura zawiera 104 klawisze, z wyodrębnionymi blokami funkcyjnym i numerycznym. Urządzenie przypominające mysz (ang. Trackball) znajduje się w prawym dolnym rogu klawiatury. Rozdzielczość zapewniana przez nie wynosi 200 punktów na cal. Jeśli chodzi o jego oprogramowanie to jest ono zgodne ze standardem Microsoft Mouse.

SZYBSZY KOPROCESOR DO AT

Monopol firmy Intel, produkującej procesory różnego typu do komputerów serii IBM PC, został złamany przez bardzo ciekawy produkt firmy Integrated Information Technology Incorporation. Jest nim układ IIT-2C87, czyli zgodny sprzętowo i programowo z modelem 80287 koprocesor numeryczny do komputerów klasy AT 286.

Kość została wykonana w technologii CMOS i pozwala na dwukrotny wzrost mocy obliczeniowej systemu. W odróżnieniu od elementów firmy Intel, dostępne są wersje koprocesora pracujące z częstotliwością 20 MHz. Koszt takiego układu $239 przy zakupie 1000 sztuk.

Dla użytkowników komputerów IMB PC AT narzekających na powolność swoich systemów w stosunku do wcześniej produkowanych XT, koprocesor firmy IIT jest rozwiązaniem z wyboru.

SOTA 386 SI



Masz starzejące się XT, znajomi z nowszymi komputerami mówią Ci, że Twój sprzęt to już historia. Słyszysz słowa OS/2, Unix, wielodostęp i zaczynasz się zastanawiać co zrobić ze swoim starym i poczciwym pudłem.

Jest rada — nie musisz go sprzedawać — wystarczy jeśli nabędziesz naszą kartę SOTA 386si, która zamieni Twój komputer w 19 razy szybszy 32-bitowy sprzęt.

Tak mogłaby wyglądać reklama kalifornijskiej firmy SOTA, zajmującej się produkcją różnego rodzaju akceleratorów do komputerów IBM PC. Omawiana karta o symbolu SOTA 386si, zawiera 32-bitowy procesor 80386SX pracujący z zegarem o częstotliwości 16 MHz. Dodanie koprocesora 80387SX zwiększa 5-krotnie numeryczne możliwości zestawu, w stosunku do zwykłego AT, wyposażonego w koprocesor 80287. Oprócz szybkości i całkowitej zgodności z najnowszym oprogramowaniem opisywana karta otwiera również drogę do programów pisanych specjalnie na 32-bitowe procesory serii 386. Dodatkowym wyposażeniem jest karta pamięci 8MB RAM. Przy takim rozszerzeniu nie mamy już żadnych problemów przy pracy z systemem operacyjnym IBM OS/2, nakładkami typu Windows/386, czy po prostu z UNIX'em, który jest niezwykle pamięciożerny.

Nie namawiam nikogo do kupowania XT z taką kartą, ale dla wielu aktualnych posiadaczy samego komputera IBM PC/XT, karta taka może stanowić ciekawą alternatywę rozwojową.

DAVID



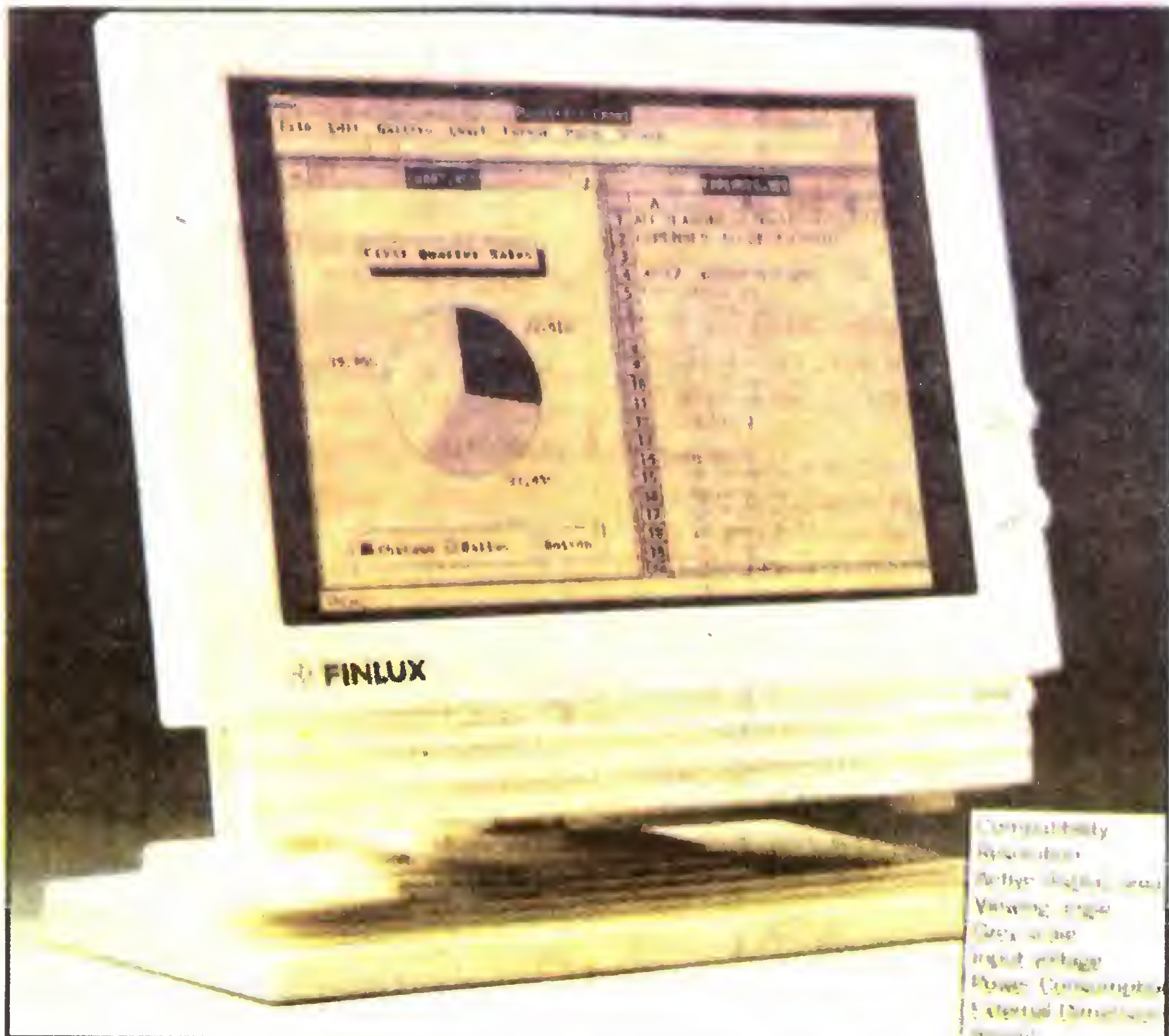
Ten młody człowiek na zdjęciu obok to David Whittaker, osoba znana każdemu posiadaczowi chipa AY do Spectrum. David jest autorem muzyki do gier Platoon, Tetris, License to Kill i wielu innych. Ostatnio firma Miles Gordon Technology podpisała z Davidem kontrakt dotyczący produkcji oprawy dźwiękowej do programów na nowy komputer Sam Coupe, jak również produkcji programów muzycznych. Przypominamy, że Sam wyposażony jest w generator dźwięków SAA 1099 o sześciu kanałach stereo. Wszyscy miłośnicy muzyki komputerowej patrzą na Davida, który z pewnością wykorzysta ten muzyczny potencjał rzucając ich na kolana.

(Gen)

PROCESORY SERII 88000 STANDARTEM DLA ELEKTRONIKI WOJSKOWEJ

Francuska firma Thomson-CSF, jeden z największych producentów wyposażenia elektronicznego dla armii, zapowiedziała standaryzację swojej produkcji w oparciu o najnowszą serię procesorów Motorola 88000. Thomson uzyskał zgodę na produkcję podstawowych układów 88100 i 88200 w wersji militarnej.

Z kolei amerykańska firma Advanced Computers doniosła o wyprodukowaniu komputera zawierającego ponad 500 procesorów serii 88000



PŁASKI EKRAN DO PC

Jeśli znudziło Cię wielkie i ciężkie pudło monitora, przed którym pracujesz, narzekasz na zmęczenie wzroku, boisz się promieniowania rentegenowskiego, to pomyśl o zakupie nowego, elektroluminescencyjnego monitora firmy FINLUX.

Ważny on mniej niż 1.5 kg, pobiera 25 W i ma rozmiary 250*200*70 (mm), czyli jest lekki, płaski i oszczędny. Realizuje standart graficzny EGA, zapewniając rozdzielczość 640*350 w trzech odcieniach szarości.

KOMPUTER TYPU LAPTOP Z KARTĄ I WYŚWIETLACZEM VGA

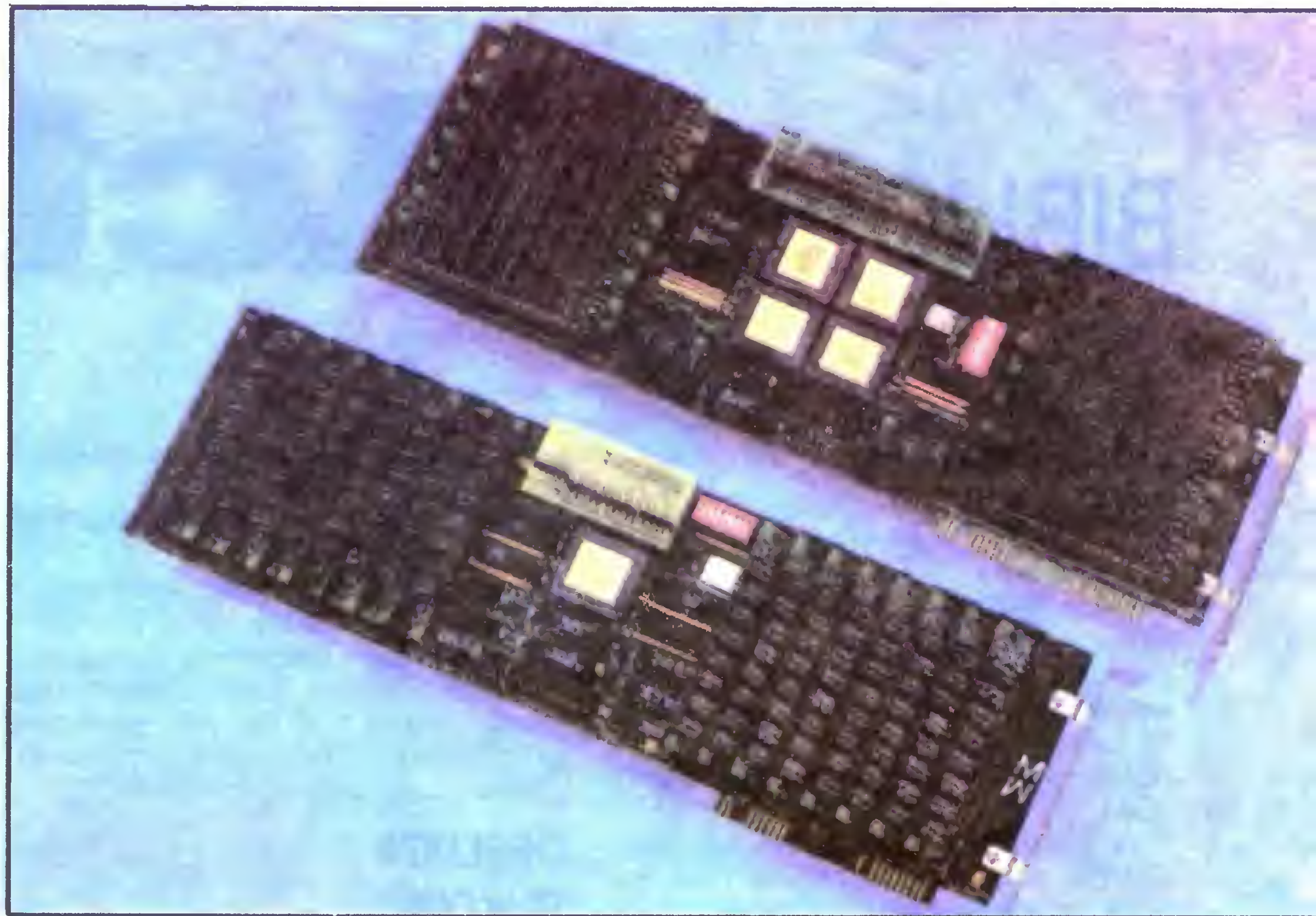


Tajwańska firma Aquarius Systems wypuściła ostatnio na rynek przenośny komputer klasy AT wyposażony w wyświetlacz i kartę standartu VGA. Oprócz charakterystycznej dla tego trybu rozdzielczości 640*480 (w 16 odcieniach szarości) zachowywano pełną zgodność ze standartami CGA, Hercules i EGA. Zastosowany wyświetlacz typu LCD (ang. Liquid Crystall Display) jest odłączalny od komputera — przy pracy stacjonarnej możemy korzystać z pełnowymiarowego monitora kolorowego.

Model Auarius ASI-168 wyposażony jest w procesor 80286·pracujący z częstotliwością 16 Mhz. Dostępna pamięć RAM wynosi 1 MB i może być zwiększona 5-krotnie. Oprócz stacji 3.5 cala o pojemności 1.44 MB zastosowano dysk twardy 20 MB o czasie dostępu 27 ms. Jeden z portów rozszerzających emuluje rozszerzenie dla komputerów Toshiba T3100. Standartowym wyposażeniem są dwa złącza RS 232 C i dwukierunkowy interface typu Centronisc, umożliwiający także podłączenie drugiego napędu dyskowego. Dodatkowe opcje to koprocesor numeryczny 80287, płyta 4MB RAM, dyski twarde 40,100 i 200 MB oraz bateria akumulatorów niklowo-kadmowych.

Całość waży 5 kg i kosztuje w wersji podstawowej $4500.

MicroMagazyn przygotował Jonasz Mayer



MULTIPUTER

Miesiąc temu starałem się Wam wytłumaczyć co to jest przetwarzanie równoległe. Żeby udowodnić, że na teorii sprawa się nie kończy, dziś przedstawię Wam istniejący komputer (a właściwie multiputer), realizujący w praktyce to o czym pisałem.

Cegłą,

z której został zbudowany MultiPuter firmy MicroWay, o którym będzie mowa, jest transputer T800. Jest to 32-bitowy transputer przystosowany do pracy z zegarem 20 MHz, oparty na architekturze RISC (czyli dysponujący zestawem niewielu, za to bardzo szybkich instrukcji). Tym, co różni transputer od zwykłego mikroprocesora są szybkie łącza, w jakie jest wyposażony. T800 ma cztery kanały szeregowe, pozwalające na przesyłanie informacji z szybkością 20 Mbit/s. Pozwala to łączyć transputery w łańcuchy (każdy transputer jest połączony z dwoma innymi) lub dwuwymiarowe sieci (możliwe są i inne kombinacje).

Ściana,

czyli prefabrykat z cegieł, to płyta zawierająca kilka transputerów — 1 (MonoPuter), 2 (BiPuter), 4 (QuadPuter), 6 (HexPuter) lub 9 (Fast 9 *Special*), każdy z kilkoma układami pozwalającymi na zarządzanie własną pamięcią RAM (do 16 Mb). Nie wszystkie transputery znajdujące się na płycie muszą być wykorzystane, chociaż biorąc pod uwagę ich ceny (od 2000 $ w górę) byłoby to marnotrawstwem.

Budynek,

czyli cały MultiPuter, składa się z dużej (nietypowej jak chodzi o rozmiary) obudowy od komputera AT 386, zawierającej solidny zasilacz (około 1 kW), komputer AT z procesorem 80386, kartę grafiki EGA, twardy dysk oraz jedenaście wolnych szyn 16-bitowych na karty rozszerzające. W te szyny wkłada się „ściany" — tyle na ile nas stać, następnie fachowiec łączy transputery między sobą kabelkami (co wcale nie jest takie proste) i cała konstrukcja jest gotowa.

Elewacja.

Dla nie wtajemniczonego użytkownika MultiPuter niczym się nie różni od zwykłego komputera z systemem MS-DOS. Transputery — o ile nas nie interesują — możemy zignorować, i korzystać z tego komputera tak samo jak z każdego innego. Popełnimy wtedy bardzo duży błąd. MultiPuter nie jest komputerem AT do którego dołączono kilka transputerów. Jest dokładnie odwrotnie — to z czym mamy do czynienia to komputer zbudowany z transputerów, w którym AT pełni rolę portiera.

Wchodzimy do środka.

Korzystanie z tego cudu techniki jest trochę dziwne, gdyż trzeba to robić za pośrednictwem procesora 80386, a nie bezpośrednio. Związane jest to z brakiem (narazie) systemu operacyjnego korzystającego z T800 jako procesora. Brak jest również innego typu oprogramowania, toteż użytkownicy są zdani na programy pisane samodzielnie. Jak je uruchomić? Nie jest to zbyt skomplikowane, aczkolwiek wymaga opanowania któregoś z języków dostosowanego do pisania programów przeznaczonych do przetwarzania równoległego. Może to być na przykład Parallel C. Korzystając z oprogramowania działającego pod kontrolą MS-DOS-u przygotowujemy treść programu, kompilujemy go i wykonujemy wszystkie inne operacje potrzebne do przygotowania gotowego kodu wynikowego przeznaczonego do pracy w MultiPuterze. Następnie przy pomocy krótkiego programu o nazwie RUN ładujemy gotowy program do sieci transputerów w której zostanie wykonany — i czekamy na wyniki.

Szybkość

zależy od ilości transputerów którymi dysponujemy. Według danych firmy MicroWay jej Model 190 MultiPuter z 19 transputerami daje 180 MIPS, czyli milionów instrukcji na sekundę (Millions Instructions Per Second). 80386, skądinąd bardzo szybki procesor, osiąga w najlepszym wypadku 4 MIPS. Możliwe jest skonstruowanie z istniejących już elementów systemu z 1000 MIPS — szybszego od niejednego Cray-a. Oczywiście są to dane teoretyczne — w praktyce osiągnięcie takiej szybkości jest nierealne.

Wąskie gardło.

Transputery między sobą komunikują się błyskawicznie, ale ich łącza nie bardzo nadają się do bezpośredniego połączenia z otoczeniem. Żeby ominąć ten problem, zaprojektowano i wykonano specjalny interfejs, łączący sieć transputerów z resztą świata. Interfejs korzysta z wolnego, 8-bitowego łącza równoległego, pracującego z szybkością 20 kB/s. Konstruktorzy wyszli bowiem z założenia, że większość użytkowników potrzebujących superszybkiego komputera ma danych niewiele, za to musi na nich wykonać dużo operacji. Dla tych którzy danych mają dużo, jest przygotowywany transputerowy kontroler twardego dysku, ale zanim będzie on gotowy trzeba relatywnie długo czekać na przekazanie danych do programu działającego w MultiPuterze i na przekazanie wyników.

Podobne kłopoty dotyczą komunikacji z kartą EGA, ale ta sprawa doczekała się już swojego rozwiązania. Istnieje bowiem specjalna karta o nazwie VideoPuter, w której transputer będący częścią sieci ma równocześnie dostęp do pamięci obrazu. Dzięki temu nie musimy w trakcie tworzenia grafiki tracić czasu potrzebnego na skomunikowanie się sieci z systemem za pośrednictwem powolnego łącza.

Dla kogo?

Cena MultiPutera (w najtańszej, czyli najskromniejszej wersji) wynosi 29000 £. Z drugiej strony dla wielu użytkowników szybkość kilkuset MIPS jest nie do pogardzenia, takimi możliwościami nie zawsze dysponują szybkie komputery typu workstation. Jak zwykle użytkownicy stają przed dylematem — kupić komputer szybszy, czy tańszy. W Polsce jak narazie takiego dylematu nie mamy — dla wielu potencjalnych użytkowników (nie tylko prywatnych), 700 dolarów na zakup komputera klasy XT jest kwotą absolutnie nieosiągalną, co dopiero mówić o dziesiątkach tysięcy na MultiPuter. Jedyne co możemy robić, żeby nie zostać całkiem w tyle, to czytać o tym, czego używają inni — i marzyć.

Marcin Borkowski

Powyższy tekst powstał na podstawie materiałów nadesłanych nam przez europejski oddział firmy MicroWay, za co serdecznie dziękujemy.

BIBLIOTEKA Action!

PROCEDURY WEJŚCIA

Teraz zajmiemy się procedurami, których działanie jest odwrotne do działania **Print** i **Put**. Służą one do wprowadzania informacji i mają zbliżony format oraz podobne warianty. Procedurami wejścia w **Action!** są: **Input** i **Get**.

W zależności od rodzaju odczytywanych danych rozróżniamy dwa rodzaje procedur **Input**: liczbowe i tekstowe. Procedura **GetD** jest natomiast tylko jedna.

Odczyt liczb

Wymienione poniżej sześć funkcji pozwala na odczytanie liczb dowolnego typu z dowolnego kanału. Działanie ich jest bardzo podobne i można je wszystkie opisać razem.

formaty: **BYTE FUNC InputB()**
BYTE FUNC InputBD(BYTE kanał)
CARD FUNC InputC()
CARD FUNC InputCD(BYTE kanał)
INT FUNC InputI()
INT FUNC InputID(BYTE kanał)

parametry: **kanał** jest poprawnym numerem IOCB (0—7)

Działanie tych funkcji jest następujące:

InputB wprowadza z ustalonego kanału IOCB liczbę BYTE.
InputBD wprowadza ze wskazanego kanału IOCB liczbę BYTE.
InputC wprowadza z ustalonego kanału IOCB liczbę CARD.
InputCD wprowadza ze wskazanego kanału IOCB liczbę CARD.
InputI wprowadza z ustalonego kanału IOCB liczbę INT.
InputID wprowadza ze wskazanego kanału IOCB liczbę INT.

Odczyt ciągów

Wprowadzanie ciągów jest wykonywane przez procedury o nazwie **„InputS"**. W bibliotece **Action!** są trzy takie procedury i pozwalają one na odczytanie ciągu znaków z dowolnego kanału i/lub określenie maksymalnej długości wprowadzanego ciągu.

formaty: **PROC InputS(‹ciąg›)**
PROC InputSD(BYTE kanał,‹ciąg›)
PROC InputMD(BYTE kanał,‹ciąg›,BYTE max)

parametry: **‹ciąg›** jest nazwą tablicy BYTE ARRAY
kanał jest poprawnym numerem IOCB (0—7)
max jest największą dopuszczalną liczbą znaków w odczytywanym ciągu. Jeśli ciąg ma więcej znaków, to zostaje obcięty do podanej długości.

Działanie tych procedur jest następujące:

InputS wprowadza z ustalonego kanału IOCB ciąg znaków o długości nie przekraczającej 255.
InputSD wprowadza ze wskazanego kanału IOCB ciąg znaków o długości nie przekraczającej 225.
InputMD wprowadza ze wskazanego kanału IOCB ciąg znaków o długości nie przekraczającej "max".

Odczyt znaku

Do odczytu pojedynczego znaku ze wskazanego kanału służy funkcja **GetD**.

format: **CHAR FUNC GetD(BYTE kanał)**

parametry: **kanał** jest poprawnym numerem IOCB (0—7)

Funkcja ta jest używana do pobrania jednego znaku z IOCB określonego przez "kanał". Znak jest zwracany przez funkcję jako liczba odpowiadająca wartości jego kodu ATASCII.

OBSŁUGA PLIKÓW

Do procedur wejścia/wyjścia zaliczają się jeszcze procedury służące do bezpośredniej obsługi plików. Są one przeznaczone przede wszystkim do komunikacji z urządzeniami zewnętrznymi komputera: klawiaturą, magnetofonem, stacją dysków, drukarką itd.

Procedura Open

Procedura **Open** przygotowuje blok IOCB (otwiera kanał) do komunikacji z urządzeniem zewnętrznym.

format: **PROC Open(BYTE kanał,‹plik›,BYTE tryb,pom2)**

parametry: **kanał** jest poprawnym numerem IOCB (0—7)
‹plik› jest stałą tekstową (lub identyfikatorem tablicy zawierającej tą stałą) określającą urządzenie (K:, C:, D:, P: itd.), dla którego otwierany jest kanał IOCB. Stacja dysków (D:) wymaga ponadto nazwy pliku.
tryb jest liczbą określającą tryb dostępu do otwieranego kanału:
4 — tylko odczyt
6 — odczyt katalogu dyskietki
8 — tylko zapis
9 — zapis na końcu pliku
12 — zapis i odczyt (wymiana)
pom2 jest wartością pomocniczą, zależną od rodzaju urządzenia (zwykle zero).

Procedura ta otwiera wskazany kanał IOCB do komunikacji z urządzeniem określonym przez ‹plik›. Przy otwieraniu konieczne jest określenie trybu dostępu do urządzenia podczas komunikacji. Zmiana tego trybu jest możliwa tylko przez zamknięcie i ponowne otwarcie kanału.

UWAGA: Nie można otwierać kanału 7, ponieważ jest on wykorzystywany przez system **Action!** jako kanał wejściowy z klawiatury. Można natomiast używać kanału 7 do wprowadzania znaków z klawiatury (K:) bez jego otwierania. **Action!** automatycznie otwiera kanał 7 przy uruchamianiu systemu.

Procedura Close

Procedura **Close** zamyka kanał komunikacji z urządzeniem zewnętrznym.

format: **PROC Close(BYTE kanał)**

parametry: **kanał** jest poprawnym numerem IOCB (0—7)

Zamknięcie wskazanego kanału kończy komunikację z urządzeniem poprzez ten kanał. Na końcu programu należy zawsze zamknąć wszystkie kanały otwarte przez program, gdyż **Action!** — w przeciwieństwie do Basica — sam nie zamyka kanałów IOCB.

Procedura XIO

format: **PROC XIO(BYTE kanał,0,rozkaz,pom1,pom2,‹plik›)**

parametry: **kanał** jest poprawnym numerem IOCB (0—7)
rozk jest kodem rozkazu wykonywanej operacji (rejestr ICCMD w IOCB).
pom1 jest pierwszym bajtem pomocniczym IOCB (rejestr ICAX1).
pom2 jest drugim bajtem pomocniczym IOCB (rejestr ICAX1).
‹plik› jest stałą tekstową określającą urządzenie (dla stacji dysków — D: — konieczna jest także nazwa pliku).

Procedura ta jest wywołaniem systemowej procedury I/O i służy głównie do wykonywania specjalnych operacji dyskowych. Jej parametry odpowiadają dokładnie parametrom stosowanym w instrukcji XIO występującej w Atari Basic. Dla niektórych podanych tam operacji **Action!** posiada specjalne, własne procedury. **XIO** jest więc w **Action!** używane znacznie rzadziej niż w Basicu.

UWAGA: zero podane jako drugi parametr jest niezbędne dla prawidłowego działania procedury.

Procedura Note

Zwraca aktualne wartości wskaźnika odczytu/zapisu w pliku dyskowym.

format: **PROC Note(BYTE kanał, CARD POINTER sektor, BYTE POINTER bajt)**

parametry: **kanał** jest poprawnym numerem IOCB (0—7)
sektor jest wskaźnikiem zmiennej, która przechowuje numer sektora.
bajt jest wskaźnikiem zmiennej, która przechowuje numer bajtu w sektorze.

Procedura umieszcza w określonych przez wskaźniki zmiennych adres (numer sektora i bajtu w sektorze) bajtu, który będzie zapisywany lub odczytywany jako następny (czyli zwraca wartość wskaźnika pliku dyskowego).

Procedura Point

Ustala wartość wskaźnika odczytu/ /zapisu w pliku dyskowym.

format: **PROC Point(BYTE kanał, CARD sektor, BYTE bajt)**

parametry: **kanał** jest poprawnym numerem IOCB (0—7)
sektor jest poprawnym numerem sektora (1/—720 lub 1—1040).
bajt jest poprawnym numerem bajtu w sektorze.

Ta procedura pozwala na ustawienie wskaźnika pliku dyskowego w dowolnym miejscu tego pliku, przez co umożliwia korzystanie z plików o dostępie swobodnym. Dla poprawnego działania procedury **Point** plik dyskowy musi być otwarty w trybie 12 (wymiana).

Wojciech Zientara

TETRIS

Gra zręcznościowo-logiczna TETRIS, napisana przez W. Gierasimowa i A. Padżitnowa z Uniwersytetu Moskiewskiego, stała się prawdziwym przebojem na całym świecie, oczywiście również w Polsce.

Została pierwotnie stworzona na komputery standardu PC wykorzystując jedynie tryb znakowy. Dystrybutor programu na rynku zachodnim, firma Spectrum Holobyte, opracowała program dodając grafikę. Dzięki doskonałemu pomysłowi i dużej „wciągalności", **TETRIS** może stanowić wzór dla twórców gier komputerowych, którzy jakoś nie mogą wyjść ze schematu „strzelaj do wszystkiego, co się rusza" bądź „weź zielony kamień i uderz goblina mieczem".

Oczywiście powstało wiele wersji **TETRIS** na różne maszyny, od Spectrum po Amigę i MacIntosha. Ostatnio pojawiła się wersja pod nazwą THE WARSAW TETRIS przeznaczona na Atari XL/XE, a napisana przeze mnie wraz z Arkiem Łukszo i ozdobiona znakomitą melodyjką autorstwa Kuby Husaka. Wykorzystuje ona grafikę PMG, okna graficzne, przerwania Display List, zawiera listę najlepszych wyników itd. Dla tych jednak, których zadowoli wersja uboższa, bez tzw. „bajerów". przedstawiam poniższy program napisany w języku Action! (podobnie jak pierwowzór).

Gra się bardzo prosto. pchnięcie joysticka w lewo lub w prawo powoduje przesunięcie klocka, pchnięcie w dół — zrzucenie klocka. Obrót klocka następuje po naciśnięciu FIRE.

A teraz nieco objaśnień. Zasadniczą częścią procedury Tetris() są trzy pętle. Pierwsza z nich, najbardziej zagnieżdżona, czterokrotnie bada stan joysticka lub przycisku FIRE. Wywołuje odpowiednie funkcje by sprawdzić, czy możliwe jest przesunięcie lub obrót, oraz przemieszcza dany klocek, jeżeli wynik testu jest pozytywny. Pętla ta wykonuje się tylko wtedy, gdy zmienna „drop" nie jest ustawiona — zmienna ta przyjmuje wartość 1 tylko wtedy, gdy wykryto pchnięcie joysticka w dół. Druga pętla, sterowana zmienną „spadl" wykonuje się dopóty, dopóki klocek nie spadnie lub nie zostanie ustawiony. W tym momencie przy pomocy funkcji suma() program sprawdza, czy jakaś linia nie jest wypełniona i „obsuwa" zawartość studni nad tą linią dzięki procedurze obsun(). Obsuwane są w ten sposób wszystkie wypełnione linie; co 10 linii wzrasta poziom trudności (zmienna lev), czemu towarzyszą proste efekty graficzne i dźwiękowe. Wreszcie trzecia, główna pętla, sterowana jest zmienną „zatetris", która przyjmuje wartość 1, gdy ulegniemy tzw. „zatetrisieniu", czyli wtedy, gdy klocek jest ustawiony w studni natychmiast po pojawieniu się na ekranie, co oznacza po prostu koniec gry.

Procedury studnia(), pause(), bzykol() i koniec() nie wymagają wyjaśnień. Tablice „wx" i „wy" są wybierane w procedurze numkl(). W **TETRIS**-ie mamy 7 rodzajów klocków, kazdy co najwyżej w 4 położeniach, co daje w sumie 19 różnych klocków. Do ich wyświetlania służy procedura wstaw(). Warto zwrócić uwagę na sposób przypisania tablic w procedurze numkl() — odbywa się to przy pomocy wskaźników. Jest to metoda bezpieczna i skuteczna, gdyż Action! traktuje nazwy tablic jako adresy. Parametry p1, p2, p3 i p4 określają „przestrzeń życiową" danego klocka, potrzebną do jego obrotu. Podobnie virtual() służy do sprawdzenia, czy klocek można przesunąć w dół lub w bok (wartości umożliwiające przesunięcie zawarte są w zmiennych „shift" i „down", odpowiednio wybieranych przez procedurę params().

Jeżeli ktoś woli grać przy pomocy klawiatury (przyzwyczajenie z IBM-a ..), program można łatwo przerobić. Oto moja sugestia: deklarujemy zmienną globalną.

```
BYTE KBCODES=$2FC
```

KBCODES jest to rejestr-cień zawierający kod ostatnio wciśniętego klawisza, wartość 255 oznacza. że żaden nie został wciśnięty. Rejestr zachowuje wartość do chwili naciśnięcia następnego klawisza, toteż po każdorazowym odczycie będziemy tam wpisywać $FF. Można to zrobić w procedurze **pause()**: przed zasadniczą pętlą zliczającą piszemy **KBCODES=$FF**.

W procedurze **Tetris()** piszemy **joy=KBCODES** zamiast **joy=stick(0)**. Przy testowaniu wartości **joy** zamiast 11 piszemy 31 (ruch w lewo- klawisz "1"), zamiast 7—26 (ruch w prawo- klawisz "3"), zamiast 13—33 (zrzucenie- spacja), wreszcie zamiast **ELSEIF strig(0) THEN...** piszemy **ELSEIF joy=30 THEN...** Oznacza to obrót klocka przez naciśnięcie "2". Możemy grać zatem czterema palcami.

Jeżeli efekt działania programu Czytelnika nie satysfakcjonuje. może on oczywiście dokonać stosownych zmian, ale... osobiście jednak polecam THE WARSAW TETRIS.

Tomasz Konatkowski

```
;    Mini Tetris
;Tomasz Konatkowski
;-------------------
; (C) "Bajtek" 1989

MODULE

BYTE x,y,shift,down,nr,dlug,rot,
       CRSINH=$2F0,ATRACT=$4D
BYTE ARRAY
 wx1=[0 0 1 2],wy1=[0 1 1 1],
 wx2=[0 1 2 0],wy2=[0 0 0 1],
 wx3=[0 1 0 1],wy3=[0 0 1 1],
 wx4=[1 0 1 2],wy4=[0 1 1 1],
 wx5=[0 1 2 1],wy5=[0 0 0 1],
 wx6=[2 0 1 2],wy6=[0 1 1 1],
 wx7=[0 1 2 2],wy7=[0 0 0 1],
 wx8=[1 2 0 1],wy8=[0 0 1 1],
 wx9=[0 1 1 2],wy9=[0 0 1 1],
 wx10=[0 1 2 3],wy10=[0 0 0 0],
 wx11=[0 0 1 0],wy11=[0 1 1 2],
 wx12=[1 0 1 1],wy12=[0 1 1 2],
 wx13=[0 1 0 0],wy13=[0 0 1 2],
 wx14=[0 1 1 1],wy14=[0 0 1 2],
 wx15=[0 0 0 1],wy15=[0 1 2 2],
 wx16=[1 1 0 1],wy16=[0 1 2 2],
 wx17=[0 0 1 1],wy17=[0 1 1 2],
 wx18=[1 0 1 0],wy18=[0 1 1 2],
 wx19=[0 0 0 0],wy19=[0 1 2 3]
INT p1,p2,p3,p4
BYTE ARRAY wx,wy


PROC studnia()

Graphics(3+16) color=3
Plot(15,0) DrawTo(15,23)
DrawTo(26,23) DrawTo(26,0)
RETURN


PROC pause(CARD x)

CARD i
FOR i=0 TO x DO OD
RETURN


PROC numkl(BYTE jaki)
 CARD POINTER zux,zuy

IF jaki=0 THEN dlug=1
  zux=@wx3 zuy=@wy3  nr=3


ELSEIF jaki=1 THEN rot=(rot+1) MOD 2
  IF rot=0 THEN dlug=3
     zux=@wx10 zuy=@wy10 nr=10
  ELSE dlug=0
     zux=@wx19 zuy=@wy19 nr=19
  FI
ELSEIF jaki=2 THEN rot=(rot+1) MOD 2
  IF rot=0 THEN dlug=2
     zux=@wx8 zuy=@wy8 nr=8
  ELSE dlug=1
     zux=@wx17 zuy=@wy17 nr=17
  FI
ELSEIF jaki=3 THEN rot=(rot+1) MOD 2
  IF rot=0 THEN dlug=2
     zux=@wx9 zuy=@wy9 nr=9
  ELSE dlug=1
     zux=@wx18 zuy=@wy18 nr=18
  FI
ELSE
  rot=(rot+1) MOD 4
  IF jaki=4 THEN
   IF rot=0 THEN dlug=2
      zux=@wx1 zuy=@wy1 nr=1
   ELSEIF rot=1 THEN dlug=1
      zux=@wx13 zuy=@wy13 nr=13
   ELSEIF rot=2 THEN dlug=2
      zux=@wx7 zuy=@wy7 nr=7
   ELSE dlug=1
      zux=@wx16 zuy=@wy16 nr=16
   FI
  ELSEIF jaki=5 THEN
   IF rot=0 THEN dlug=2
      zux=@wx2 zuy=@wy2 nr=2
   ELSEIF rot=1 THEN dlug=1
      zux=@wx14 zuy=@wy14 nr=14
   ELSEIF rot=2 THEN dlug=2
      zux=@wx6 zuy=@wy6 nr=6
   ELSE dlug=1
      zux=@wx15 zuy=@wy15 nr=15
   FI
  ELSE
   IF rot=0 THEN dlug=2
      zux=@wx4 zuy=@wy4 nr=4
   ELSEIF rot=1 THEN dlug=1
      zux=@wx11 zuy=@wy11 nr=11
   ELSEIF rot=2 THEN dlug=2
      zux=@wx5 zuy=@wy5 nr=5
   ELSE dlug=1
      zux=@wx12 zuy=@wy12 nr=12
   FI
  FI
FI
wx=zux^ wy=zuy^
IF dlug=0 THEN p1=-2 p2=0 p3=2 p4=3
ELSEIF dlug=1 THEN
   p1=-1 p2=0 p3=2 p4=2
ELSEIF dlug=2 THEN
   p1=-1 p2=0 p3=2 p4=2
ELSE p1=0 p2=-2 p3=3 p4=1
FI
RETURN


PROC wstaw(BYTE x,y,kol)
 BYTE i

FOR i=0 TO 3
DO
 color=kol
 Plot(15+x+wx(i),y+wy(i))
OD
RETURN


BYTE FUNC otocz(INT x1,y1,x2,y2)
 BYTE i,j,s

s=0
FOR i=x+15+x1 TO x+15+x2
DO
FOR j=y+y1 TO y+y2
 DO
  s==+Locate(i,j)
 OD
OD
RETURN(s)


BYTE FUNC virtual()
 BYTE i,s,a,b

s=0
FOR i=0 TO 3
DO
  a=x+15+wx(i) b=y+wy(i)
  s==+Locate(a,b)
OD
RETURN(s)


PROC params(BYTE jaki)

IF jaki>10 THEN shift=2 down=4
 ELSE shift=4 down=2
FI
IF jaki=3 THEN down=4
 ELSEIF jaki=10 THEN shift=6 down=0
 ELSEIF jaki=19 THEN shift=0 down=6

FI
RETURN


BYTE FUNC suma(BYTE linia)
 BYTE i,s

s=0
FOR i=16 TO 25
DO
 s==+Locate(i,linia)
OD
RETURN(s)


PROC bzykol()
 CARD i

FOR i=30 TO 38 step 3
DO
 Sound(0,220-i,6,8)
 pause(45)
 Sound(0,92,10,10)
 pause(330)
OD
Sndrst()
RETURN


PROC obsun(BYTE linia)
 BYTE i,r

r=linia
while (linia>0)
DO
 FOR i=16 TO 25
 DO
  IF Locate(i,linia-1)=2 THEN color=2
   ELSE color=0
  FI
  Plot(i,linia)
  IF linia=r THEN bzykol() FI
 OD
 linia==-1
OD
color=0 Plot(16,0) DrawTo(25,0)
RETURN


PROC koniec(CARD pts,lev)

Graphics(0) CRSINH=1
PutE()
PrintF("Zdobyles %d pkt.",pts)
PutE()
PrintF("na poziomie %d !!!",lev)
pause(10000)
CRSINH=0
RETURN


PROC Tetris()
 CARD time,opoz,pts,temp
 BYTE jaki,i,j,joy,koliz,drop,spadl,
        sx,sy,rzad,zatetris,lin,lev

time=7000 temp=time
zatetris=0 pts=0 lin=0 lev=0
studnia()
Plot(3,0)

DO
  rot=0 drop=0 spadl=0
  time=temp  opoz=2500
  pause(time+opoz)
  jaki=Rand(7)
  numkl(jaki) params(nr)
  x=5 y=0
  IF nr=10 THEN y=2 FI
  sx=x sy=y
  wstaw(x,y,2) pts==+2
  DO
   FOR j=1 TO 4
   DO
   IF drop=1 THEN EXIT FI
   joy=Stick(0)
   IF joy=11 THEN sx=x x==-1
      IF virtual()=shift THEN
       wstaw(sx,y,0) wstaw(x,y,2)
      ELSE x=sx
      FI
   ELSEIF joy=7 THEN sx=x x==+1
      IF virtual()=shift THEN
       wstaw(sx,y,0) wstaw(x,y,2)
      ELSE x=sx
      FI
   ELSEIF joy=13 THEN
      temp=time  time=0
      opoz=0  drop=1 EXIT
   ELSEIF strig(0)=0 THEN
        koliz=otocz(p1,p2,p3,p4)
        IF koliz=8 THEN
           wstaw(x,y,0)
           numkl(jaki)
           params(nr)
           IF nr=19 THEN
              x==+2 y==-1 FI
           IF nr=10 THEN


              x==-2 y==+1 FI
           wstaw(x,y,2)
           pause(1000)
        FI
   FI
   pause(time)
   OD
  sy=y y==+1
  IF virtual()=down THEN
   wstaw(x,sy,0) wstaw(x,y,2)
  ELSE spadl=1
    rzad=y+2-dlug
    joy= (22-y)/4  pts==+joy
    WHILE (rzad>y-2)
    DO
     IF suma(rzad)=20 THEN
       obsun(rzad)
       ATRACT=0    pts==+50   lin==+1
       IF (lin MOD 10 =0)
          and (lev<10) THEN
        lev==+1
        color=2 Plot(3,lev*2)
        Sound(0,123,12,6)
        pause(10000)
        Sndrst()
        pts==+100 temp==-500
       FI
       y==-1
     ELSE rzad==-1
     FI
    OD
    EXIT
  FI
  UNTIL spadl
  OD
  IF y=1 THEN zatetris=1 FI
UNTIL zatetris
OD
koniec(pts,lev)
RETURN


PROC main()
 BYTE a

DO
 Tetris()
 PutE() PutE()
 PrintF("Jeszcze raz? [T/N]")
 Open(1,"K:",4,0)
 a=GetD(1)
 Close(1)
UNTIL a='n OR a='N
OD
```

SYNTEZATOR

Program jest napisany na komputerze Atari 800XL. Jest stosunkowo krótki, wczytuje się przez około 20 jednostek licznika. Możliwości programu są następujące:

— 4 rodzaje brzmień;

— regulowana szybkość opadania dźwięku;

— jednocześnie trzy oktawy na klawiaturze;

— dość szybka praca pomimo użycia Atari Basic.

Po uruchomieniu programu wybieramy numer brzmienia oraz szybkość opadania dźwięku i już można grać. Gra się na klawiszach od ‹TAB› do ‹RETURN› i od ‹Z› do ‹,›. Odpowiednio nad nimi umieszczone są półtony. Początki oktaw są przypisane klawiszom: ‹TAB› — dolne C, ‹U› — środkowe C i ‹Z› — górne C. Klawiszem ‹START› powraca się do wyboru parametrów.

Piotr Bendyk
Rumia

Nadesłany program wykonuje postawione zadanie i to dosyć dobrze. Jest jednak napisany bardzo nieporządnie — wystarczy przyjrzeć się numeracji wierszy. To jest — w zasadzie — drobiazg, choć wygląda nie najlepiej. Natomiast poważne błędy są gdzie indziej (na szczęście w tym przypadku nie mają one wpływu na poprawną pracę programu). Proszę przyjrzeć się wierszom 212, 230, 288. Znajdujące się tam instrukcje RETURN powodują zakończenie procedury, ale pozostawiają niezamknięte pętle FOR/NEXT. W każdym z tych wierszy instrukcja RETURN powinna być poprzedzona przez POP. Nieco mniejszy błąd znajduje się w wierszu 250. Nie przerywa on pracy programu tylko dlatego, że niweluje go instrukcja w następnym wierszu. Błąd ten jest spowodowany brakiem instrukcji RETURN po NEXT U.

Proponowałbym także wprowadzenie do programu kilku zmian, które zwiększą jego atrakcyjność. Teraz wykorzystywany jest tylko jeden z klawiszy konsoli. Można przecież użyć wszystkich. Na przykład klawisz ‹START› wybiera niższe brzmienie (B=B–1), klawisz ‹OPTION› wyższe (B=B+1), zaś klawisz ‹SELECT› zmienia wartość brzmienia o dwa (B=B+2). Oczywiście za każdym razem trzeba sprawdzać, czy wartość B znajduje się w dopuszczalnym zakresie i odpowiednio ją korygować. Jeśli po zwiększeniu uzyskamy B=5, to trzeba zmienić na B=1 i w ten sposób otrzymujemy możliwość zmiany brzmienia bez konieczności przerywania gry. Podobnie przez wykorzystanie klawiszy ‹+› i ‹*› można płynnie zmieniać czas opadania dźwięku. Ostatni z klawiszy konsoli — ‹HELP› — można użyć do przerywania pracy programu. Ze względu na zablokowanie klawisza ‹BREAK› jest to obecnie możliwe tylko przy użyciu ‹RESET›. Sposób taki jest mało elegancki i każdy przyzwoity program powinien umożliwiać jego zakończenie bez wyłączania komputera i używania ‹RESET›.

Wymienione wyżej zmiany spowodują zmniejszenie prędkości działania programu. Można tego uniknąć przez przeniesienie na koniec części inicjującej (wiersze 4–25) i instrukcji DATA. Także przeniesienie zasadniczej pętli programu (wiersze 30–45) za procedury tworzenia dźwięku przyspieszy działanie programu. Więcej informacji na ten temat można znaleźć w artykule „Szybki, szybszy, Atari" opublikowanym w „Bajtku" 3/89.

Wojtek

LISTING 1

```
XR   0 REM SYNTEZATOR XTD
OC   1 REM Piotr Bendyk
XZ   2 REM Copyrigth (C) Bajtek
NI   3 REM
KM   4 POKE 566,158
RH   5 DIM A(255):RESTORE :TRAP 18:FOR U=0
     TO 255:A(U)=0:NEXT U
BM   6 READ B,C:A(B)=C:GOTO 6
VV   10 DATA 44,243,47,217,46,193,42,182,40
     ,162,45,144,43,128,11,121,13,108
KC   11 DATA 8,96,10,91,14,81,15,72,12,64,3
     1,230,30,204,24,173,29,153,27,136
NJ   12 DATA 53,114,48,102,54,85,55,76,52,6
     8,23,60,22,53,18,47,16,45,21,40
AX   13 DATA 35,35,37,31,32,29,62,57,56,50,
     61,42,57,37,1,33,ERR,ERR
FF   18 GRAPHICS 2+16:SETCOLOR 0,1,15:POSIT
     ION 5,2:? #6;"SYNTEZATOR":POSITION 8,5
     :? #6;"xtd":SETCOLOR 1,12,5
ZF   19 FOR U=1 TO 1000:NEXT U
ZS   20 TRAP 20:GRAPHICS 0:SETCOLOR 2,0,0:S
     ETCOLOR 1,0,15:POKE 752,1:POKE 729,0:P
     OKE 731,0
YN   22 ? :? :? "PODAJ BARWE BRZMIENIA [1-4
     ]: ";:POKE 764,255:INPUT B:IF B<1 OR B
     >4 OR B<>INT(B) THEN 20
EU   23 ? CHR$(125):? :? "PODAJ SZYBKOSC OP
     ADANIA DZWIEKU:";:INPUT SP
PY   24 ? CHR$(125):POSITION 6,5:? "START -
      ZMIANA PARAMETROW"
NG   25 POKE 729,1:POKE 730,1:POKE 731,255
TA   30 X=PEEK(764):IF PEEK(53279)=6 AND PE
     EK(764)=255 THEN 20
BD   31 IF A(X)=0 THEN SOUND 0,0,0,0:SOUND
     1,0,0,0:GOTO 30
DB   35 IF X=255 THEN 30
NK   40 ON B GOSUB 200,220,250,280
SN   45 GOTO 30
JM   200 POKE 764,255:SOUND 0,A(X),14,15:FO
     R C=1 TO 5:NEXT C:SOUND 0,A(X)+2,14,12
     :FOR C=1 TO 4:NEXT C
KF   204 IF PEEK(764)=X THEN 200
ZR   205 IF PEEK(764)<>255 THEN RETURN
FC   208 FOR U=15 TO 0 STEP -SP
SU   210 SOUND 0,A(X),14,U:FOR C=1 TO 5:NEX
     T C:SOUND 0,A(X)+2,14,U/1.2:FOR C=1 TO
      4:NEXT C
ZK   212 IF PEEK(764)<>255 THEN RETURN
ST   214 NEXT U:SOUND 0,0,0,0:RETURN
KO   220 POKE 764,255:SOUND 0,A(X),14,15:SO
     UND 1,A(X)+1,14,15
MD   222 IF PEEK(764)=X THEN 220
ZP   223 IF PEEK(764)<>255 THEN RETURN
EX   225 FOR U=15 TO 0 STEP -SP
KE   228 SOUND 0,A(X),14,U:SOUND 1,A(X)+1,1
     4,15
ZI   230 IF PEEK(764)<>255 THEN RETURN
NS   232 NEXT U:SOUND 0,0,0,0:SOUND 1,0,0,0
     :RETURN
YC   250 POKE 764,255:FOR U=A(X)-2 TO A(X)+
     2:SOUND 0,U,14,15:IF PEEK(764)=X THEN
     NEXT U
ZV   253 IF PEEK(764)<>255 THEN RETURN
FD   255 FOR U=15 TO 0 STEP -SP
EB   256 FOR V=A(X)-2 TO A(X)+2 STEP 0.6
LH   258 SOUND 0,V,14,U:IF PEEK(764)=255 TH
     EN NEXT V:NEXT U:SOUND 0,0,0,0:RETURN
AJ   259 RETURN
WK   280 POKE 764,255:SOUND 0,A(X),14,15
SP   282 IF PEEK(764)=X THEN 280
AB   283 IF PEEK(764)<>255 THEN RETURN
FJ   285 FOR U=15 TO 0 STEP -SP
QW   288 SOUND 0,A(X),14,U:IF PEEK(764)<>25
     5 THEN RETURN
TW   289 NEXT U:SOUND 0,0,0,0:RETURN
```

ERRATA

W programie „Przewijanie napisów" opublikowanym w „Bajtku" 4/89 znalazły się dwa błędy uniemożliwiające jego poprawne działanie.

W wierszu 50 zamiast POKE 9193,0 powinno być POKE 8193,0, zaś na końcu wiersza 80 brak liczb 98 i 228.

(red)

LISTING 2

```
XR   0 REM SYNTEZATOR XTD
OC   1 REM Piotr Bendyk
XZ   2 REM Copyrigth (C) Bajtek
VT   3 REM Wersja poprawiona
NJ   4 REM
NJ   10 GOTO 600
BV   20 B=B+1:IF B=5 THEN B=1
ZT   30 RETURN
JD   40 B=B+2:IF B>4 THEN B=B-4
ZV   50 RETURN
CK   60 B=B-1:IF B=0 THEN B=4
ZX   70 RETURN
NX   100 POKE 764,255:SOUND 0,Y,14,15:FOR C
     =1 TO 5:NEXT C:SOUND 0,Y+2,14,12:FOR C
     =1 TO 4:NEXT C
IV   110 IF PEEK(764)=X THEN 100
ZF   120 IF PEEK(764)<>255 THEN RETURN
EJ   130 FOR U=15 TO 0 STEP -SP
EX   140 SOUND 0,Y,14,U:FOR C=1 TO 5:NEXT C
     :SOUND 0,Y+2,14,U/1.2:FOR C=1 TO 4:NEX
     T C
ZW   150 IF PEEK(764)<>255 THEN POP :RETURN
SQ   160 NEXT U:SOUND 0,0,0,0:RETURN
NR   200 POKE 764,255:SOUND 0,Y,14,15:SOUND
      1,Y+1,14,15
JV   210 IF PEEK(764)=X THEN 200
ZG   220 IF PEEK(764)<>255 THEN RETURN
EK   230 FOR U=15 TO 0 STEP -SP
JF   240 SOUND 0,Y,14,U:SOUND 1,Y+1,14,15
ZX   250 IF PEEK(764)<>255 THEN POP :RETURN
NS   260 NEXT U:SOUND 0,0,0,0:SOUND 1,0,0,0
     :RETURN
XG   300 POKE 764,255:FOR U=Y-2 TO Y+2:SOUN
     D 0,U,14,15:IF PEEK(764)=X THEN NEXT U
     :RETURN
ZF   310 IF PEEK(764)<>255 THEN RETURN
EJ   320 FOR U=15 TO 0 STEP -SP
XZ   330 FOR V=Y-2 TO Y+2 STEP 0.6
KI   340 SOUND 0,V,14,U:IF PEEK(764)=255 TH
     EN NEXT V:NEXT U:SOUND 0,0,0,0:RETURN
ZJ   350 RETURN
MG   400 POKE 764,255:SOUND 0,Y,14,15
LV   410 IF PEEK(764)=X THEN 400
ZI   420 IF PEEK(764)<>255 THEN RETURN
EM   430 FOR U=15 TO 0 STEP -SP
AJ   440 SOUND 0,Y,14,U:IF PEEK(764)<>255 T
     HEN POP :RETURN
SR   450 NEXT U:SOUND 0,0,0,0:RETURN
FW   500 X=PEEK(764):ON PEEK(53279) GOSUB 3
     0,30,20,30,40,60:IF PEEK(732) THEN 700
EI   510 IF X=7 THEN SP=SP-0.04:POKE 764,25
     5:IF SP<0.01 THEN SP=0.02
QP   520 IF X=6 THEN SP=SP+0.04:POKE 764,25
     5
PW   530 Y=A(X):IF Y=0 THEN SOUND 0,0,0,0:S
     OUND 1,0,0,0:GOTO 500
IL   540 IF X=255 THEN 500
RT   550 ON B GOSUB 100,200,300,400:GOTO 50
     0
YW   600 GRAPHICS 2+16:SETCOLOR 0,1,15:POSI
     TION 5,2:? #6;"SYNTEZATOR":POSITION 8,
     5:? #6;"xtd":SETCOLOR 1,12,5
BP   610 POKE 566,158:DIM A(255):RESTORE :T
     RAP 630:FOR U=0 TO 255:A(U)=0:NEXT U
MQ   620 READ B,C:A(B)=C:GOTO 620
KT   630 TRAP 630:GRAPHICS 0:SETCOLOR 2,0,0
     :SETCOLOR 1,0,15:POKE 752,1:POKE 729,0
     :POKE 731,0:B=1:SP=1
QE   640 POKE 82,6:? CHR$(125):POSITION 6,5
     :? "START - NIZSZE BRZMIENIE"
VG   650 ? :? "OPTION - WYZSZE BRZMIENIE":?
      :? "SELECT - SKOKOWA ZMIANA BRZMIENIA
     ":? :? "HELP - KONIEC PROGRAMU"
NY   660 ? :? :? CHR$(27);CHR$(158);" - SZY
     BSZE OPADANIE"
CN   670 ? :? CHR$(27);CHR$(159);" - WOLNIE
     JSZE OPADANIE"
SI   680 POKE 729,1:POKE 730,1:POKE 731,255
     :POKE 732,0:GOTO 500
AK   700 POKE 566,146:POKE 729,48:POKE 730,
     6:POKE 731,0:POKE 82,2:GRAPHICS 0:END
CK   800 DATA 44,243,47,217,46,193,42,182,4
     0,162,45,144,43,128,11,121,13,108
SJ   810 DATA 8,96,10,91,14,81,15,72,12,64,
     31,230,30,204,24,173,29,153,27,136
SJ   820 DATA 53,114,48,102,54,85,55,76,52,
     68,23,60,22,53,18,47,16,45,21,40
AC   830 DATA 35,35,37,31,32,29,62,57,58,50
     ,61,42,57,37,1,33
```

KOMPUTER Z CHLAPACZEM

Kilka lat temu nie śniło się nam nawet, że tak niedobre urządzenie, jakim jest komputer, może mieć jakikolwiek związek z motoryzacją. Obecnie w większości samochodów komputer „pokładowy" stanowi prawie że standardowe wyposażenie; okazuje się jednak, że związków komputera z motoryzacją jest znacznie więcej.

Podczas wizyty w jednym ze sklepów PEWEX-u przypadkowo zupełnie dowiedziałem się, że dwa chlapacze ze znakiem firmowym do samochodu zagranicznego kosztują jedyne 36 dolarów; jest to cena podejrzewam, całkiem rozsądna i prawdopodobnie nawet PEWEX ma na takie chlapacze zbyt, skoro są one dostępne.

Oczywiście człowiek zdrowy na umyśle nigdy danych chlapaczy nie kupi za równowartość 250000 złotych, lecz pójdzie do pierwszego lepszego sklepiku prywatnego albo zwinie sąsiadom dwie gumowe wycieraczki spod drzwi. Dorobienie odpowiednich wsporników metalowych zleci się panu Ziutkowi w państwowej tokarni za 5 tysięcy i kłopot z głowy. Żelazne prawa rynku dowiodły, że chlapacz zachodni wcale nie jest lepszy od krajowego, zwłaszcza jeśli się jeździ po tak równych drogach, jak warszawskie ulice, gdzie nierzadko można urwać cały tylny most z bagażnikiem włącznie, a nie tylko głupi chlapacz.

W tejże samej sieci sklepów komputer Commodore 128D dzięki niesłychanej łasce speców od handlu staniał obecnie do ceny 540 USD, która to cena byłaby może nawet i aktualna, ale w 1985 roku. Jakie są efekty tego kroku dla samego PEWEX-u nie wiem; wiem tylko tyle, że komputer AMIGA można kupić bez żadnych problemów na giełdzie za 400—450 USD co oznacza, że prywatny import znacznie lepiej wczuł się w sytuację, aniżeli przedsiębiorstwo w zasadzie predestynowane do robienia takich transakcji. Nie da się bowiem żadną miarą ukryć, że AMIGA jest komputerem o niebo lepszym od wspominanego C-128D i budzi obecnie znacznie więcej zachwytu, choćby ze względu na szybkość, grafikę czy nowszy procesor.

Niestety PEWEX albo AMIGI nie lubi, albo nie zdaje sobie sprawy, jak dobry interes można na niej ubić. Co prawda słyszałem plotki, że AMIGA objęta jest embargiem technologicznym, co jest o tyle wesołe, że ATARI 520ST i pokrewne niewiele różnią się od wspominanej AMIGI pod względem technologicznym, a jednak w jakiś niewątpliwie cudowny sposób docierają i zalegają półki.

Odrębną i niezmiernie frapującą sprawą są zielone monitory produkcji rodzimej z powodzeniem sprzedawane na giełdzie za 30 dolarów. Mój niekłamany zachwyt i podziw wzbudza tu fakt, że na monitorze można czasem wyświetlać różne zielonkawe napisy, a na dolarze nie. Dziwnym trafem jednak kolor obydwu jest jakby ten sam.

Nikt mi nie wmówi, że nasz przemysł elektroniczny nie wie, jak się monitory produkuje; swego czasu kosztowały one nawet drożej od telewizorów, co było najlepszym dowodem na następne prawo rynkowe, że mniejsza ilość podzespołów może kosztować znacznie drożej pomimo identycznej konstrukcji. Niezbite jest, że swego czasu monitory były, teraz ich nie ma, do czego jesteśmy od jakiegoś czasu przyzwyczajeni. Tu komputery mają przewagę nad motoryzacją, gdzie opon na przykład poza PEWEX-em lub giełdą się nie zobaczy: zastanawiam się tylko, czy ktokolwiek prowadzi w kraju statystykę, ile osób już zginęło dzięki jeździe na łysych lub uszkodzonych oponach. Takiego niebezpieczeństwa nie ma w wypadku monitorów; to, że kilkadziesiąt tysięcy młodych ludzi popsuje sobie wzrok, gapiąc się w migoczące telewizory, nikogo specjalnie nie obchodzi.

Z drugiej strony miłe jest, że monitor jest na giełdzie tańczy od pary chlapaczy; na chlapaczach dane można co najwyżej wyryć, a długotrwałe wpatrywanie się w znak firmowy może doprowadzić do niezdrowych i niesłusznych porównań i skojarzeń. Ponadto za brak chlapacza rypną Ci mandat, a za brak monitora i popsuty wzrok nie.

Mój znajomy twierdzi, że bajtodolar (symbol B$ — nie mylić ze zmienną tekstową B$!), który całkowicie zmonopolizował rynek mikrokomputerowy w kraju, nie ma szans wpływania do sakiewki PEWEX-u dopóki specom od nakładania marż i podatków nie wyświetli się komunikat STRING TOO LONG, ERROR podczas kalkulacji nowej „atrakcyjnej" ceny. Moim zdaniem nie ma obawy — oni nie mają przecież monitorów, bo i skąd.

Klaudiusz Dybowski

INSIDE COMMODORE DOS

Książka ta trafiła do moich rąk w zupełnie przypadkowy sposób — skuszony raczej ceną aniżeli potrzebą podczas wizyty w małej księgarence nabyłem ją za sumę jednego dolara i 25 centów. „Pozaginane okładki" — oświadczyła sprzedawczyni na moje zdziwienie. Pod naklejką z ceną widniała cena oryginalna — 19.95 $.

Nie tak dawno miałem okazję przedstawić Czytelnikom dwie inne publikacje tego typu — „1571 INTERNALS" i „ANATOMY OF THE 1541 DISK DRIVE", obie wydane przez firmę DATA BECKER z Dusseldorfu. „INSIDE COMMODORE DOS" jest również mapą pamięci stacji 1541 zawierającą wydruk pamięci ROM z komentarzem.

Książka ta oferuje jakby nieco bogatszy tematycznie materiał: sam wydruk zawartości ROM jest krótszy i pełniej skomentowany, przedstawiono tu również bardzo wiele gotowych do wprowadzenia programów. Dzięki temu Czytelnik, jeśli nawet nie bardzo „chwyta" temat, to za pomocą gotowego programu może sobie szybko empirycznie sprawdzić, o czym jest mowa w książce.

Tekst jest przedstawiony w sposób jasny i przejrzysty; dużą zaletą jest wyjaśnienie wielu kwestii wątpliwych, a nie omówionych w firmowej instrukcji obsługi. Przykładem mogą tu być np. polecenia B-R i B-W; większość programistów zastępuje je zwykle poleceniami równoznacznymi „U1" i „U2". Wyjaśnienie tego i wielu innych zagadek znajdziesz w tej książce.

Sam wstęp dla początkujących może się niektórym wydać zbyt krótki; z drugiej strony od tego właśnie jest instrukcja obsługi. Zaoszczędzone w ten sposób miejsce przeznaczono (z dobrym efektem) na informacje znacznie ciekawsze. Zapewne niewiele osób wie, że właśnie z tej książki pochodzą popularne swego czasu programiki do tworzenia błędów na dyskietkach (np. „23 ERROR"), „BULK ERASER 1541", „GCR-HEX" i wiele innych. Ważne jest również to, że programy te w zdecydowanej większości są dobrze omówione, dzięki czemu można nie tylko „wpalcować", ale i zrozumieć dane zagadnienie. W wielu wypadkach oprócz wersji programu w BASIC-u podane są również wydruki źródłowe programów lub procedur napisanych w języku maszynowym. Gdzieniegdzie procedury takie są również dość dokładnie objaśnione (w sensie komentarza do poszczególnych instrukcji).

Do unikalnych tematów można zaliczyć bardzo dokładne omówienie konwersji danych na format GCR. Jak wiadomo kodowanie to stanowi klucz do przyspieszenia działania stacji i tworzenia własnych programów przyspieszających (żargonowych „dopalaczy"). Oprócz tego wyjaśniono przyczyny niekompatybilności stacji 1541 (niemożność zapisu na dyskietkach) ze starszymi modelami (np. 4040), zawarto zestawienie błędów DOS V2.6, porównano dwie wersje ROM stacji — 901229-03 i 901229-05. Roztrzepani z kolei znajdą tu kilka ciekawych porad dotyczących odzyskiwania zbiorów z dyskietek uszkodzonych czy nawet sformatowanych.

Moim zdaniem „INSIDE COMMODORE DOS" jest publikacją dla raczej zaawansowanych Czytelników. Do dokładnego zrozumienia wszystkich omawianych w książce zagadnień na pewno będzie przydatna (a miejscami niezbędna) głębsza wiedza o funkcjonowaniu takich urządzeń w ogóle. 46 gotowych do wpisania programów umożliwia z kolei szybsze zapoznanie się z wyższymi poziomami wiedzy o stacji i jej pracy. Zawarte w książce programy uzupełniają się zresztą wzajemnie, tworząc dość uniwersalne narzędzie pracy i dla początkujących i dla zaawansowanych. Z mojego punktu widzenia przewodnik ten jest momentami niezastąpiony, zwłaszcza przy rozwiązywaniu nieco trudniejszych kwestii — np. programowaniu „wirusów".

Klaudiusz Dybowski

Richard Immers, Gerald G. Neufeld
„INSIDE COMMODORE DOS"
Wydawca: DATAMOST, Inc.,
Wydanie drugie z 1985
Stron 508, cena 19.95$
ISBN 0-8359-3091-2

GEOS 128

Prawdopodobnie nikt w firmie Berkeley Softworks nie przypuszczał, że GEOS będzie jednym z najlepiej sprzedających się systemów dla C-64 i C-128. W ciągu kilku lat ukazało się 5 jego wersji (cztery dla C-64 i jedna dla C-128), niebawem na rynku pojawi się najnowsze dziecko — GEOS V2.0, który sądząc po ilustracjach reklamowych gwarantuje użytkownikom zupełnie nową jakość.

Dzięki uprzejmości jednego z moich przyjaciół miałem okazję zapoznać się z wersją GEOS dla C-128. Szczerze mówiąc spodziewałem się chyba czegoś więcej. GEOS 128 pozostawił we mnie uczucie pewnego niedosytu choć dla fanów tego systemu jest on dość łakomym kąskiem.

W porównaniu z wersją V1.3 dla C-64 GEOS 128 oferuje głównie dwa ulepszenia, po pierwsze szybkość (praca w oparciu o zegar 2 MHz), po drugie możliwość (nareszcie!) korzystania z dwustronnego trybu pracy stacji 1571. Trzecią nowością jest tu praca zarówno w 40- jak i 80-znakowym trybie pracy co oczywiście znacznie polepsza czytelność i jakość grafiki. Ma to duże znaczenie przede wszystkim podczas pracy z programami GEOWRITE (graficzny edytor tekstu) i GEOPAINT (edytor graficzny).

Na kłopoty natknąłem się już podczas pierwszego uruchomienia systemu w trybie 80 znakowym. Po wczytaniu czołówki programu została zerwana synchronizacja obrazu co jak się potem okazało nie jest odosobnionym przypadkiem. Próbowałem uruchamiać GEOS 128 na kilku monitorach — zawsze z tym samym skutkiem. Oczywiście posiadaczom lepszych monitorów (z potencjometrem synchronizacji wyprowadzonym na zewnątrz) nietrudno będzie ustawić odpowiednio obraz: posiadacze rodzimych Neptunów zmuszeni jednak będą do grzebania w monitorze co nie jest ani bezpieczne ani pożądane. W każdym razie dor[illegible] bym mniej zaawansowanym wykonanie tej ope[illegible] pod troskliwym (?) okiem [illegible]tach [illegible].

Na duży plus należy zaliczyć autorom m[illegible] korzystania z większości programów [illegible] pod kontrolą GEOS na C 64. Mowa tu [illegible] o programach wydanych przez f[illegible] niektóre programy użytkowe pisane prz[illegible] rów [illegible] sympatyków nie będą działały [illegible] ograniczeniem na jakie się tu natknąłem była ko[illegible] nieczność przełączenia trybu prac[illegible] 40 znaków. Można tego dokonać albo poprzez [illegible] GEOS [illegible] opcję SWITCH 40/80 albo w chwi[illegible] chamiania systemu.

Bez żadnych problemów działają programy [illegible] jące dla drukarek programy obsługują[illegible] (joystick, mysz[illegible] pióro świetlne). W GEOWRITE możliwe jest korzystanie z zestawu [illegible] stepnego [illegible] C 64 podobn[illegible] wa wczytywanie rysunków wykonanych na [illegible] 64. W obu [illegible] [illegible] przełączanie [illegible] 80 znako[illegible] wym widzimy [illegible] tekst [illegible] rysunek.

Nieco inaczej [illegible] GEOPAINT ok[illegible] ikonam symbolizującym [illegible] kreślarskie ołówek, gumk[illegible] Pojawiają [illegible] ekranie dopiero po nałożeniu wskaź[illegible] myszk lub joysticka [illegible] ej części ekranu [illegible] przycisku. Ponowne wciśnięcie powoduje ich zniknięcie.

Znacznie ulepszono opcję RESET. W C 64 jej wybór powodował automatyczny powrót do BASIC, w C-128 natomiast powoduje ona jedynie powrót do stanu jak zaraz po wczytaniu i uruchomieniu systemu. Jest to o tyle wygodne, że w razie jakiejś wpadki (np. próby przeniesienia zbiorów pomiędzy dyskietkami o tych samych nazwach) nie trzeba wyłączać komputera i wczytywać systemu od nowa.

Wiele opcji niestety nie zostało zmienionych. Na przykład kopiowanie programów z dyskietki na dyskietkę (gdy używamy stacji 1571) wymaga wciąż tych samych dość uciążliwych operacji (wyprowadzenia ikony lub ikon na ramkę, zmiany dyskietki itd.). Biorąc pod uwagę, że na dyskietce takiej mieści się teraz 336 KB to operacja ta potrafi zająć dobre 30 minut przy przenoszeniu kilkudziesięciu zbiorów. Podobnie uporządkowanie katalogu dyskietki (przenoszenie zbiorów ze strony na stronę, kasowanie) zajmuje również sporo czasu. W programie sterującym dla myszki można było dodać procedurę obsługującą drugi, niewykorzystywany zwykle przycisk czego jednak nie uczyniono. No cóż, może następca czyli wersja V2.0 zlikwiduje pozostałe braki...

Reasumując GEOS 128 jest na pewno ulepszoną wersją systemu lecz wersją która nie wykorzystała do końca możliwości tkwiących w C-128. Oferuje ona użytkownikowi kilka istotnych zmian co jednak wiąże się raczej ze zmianą komputera aniżeli przystosowania czy modyfikacji koncepcji samego systemu.

Klaudiusz Dybowski

-WALKER-
-DEMO-
-I INNE...-

Skromne możliwości graficzne i muzyczne Amigi potrafią przyprawić o nerwowy tik posiadaczy maszyn takich jak Zx Spectrum, nawet jeśli ukochany „Spektruś" ma generator dźwięku oparty na chipie AY3-8910. Wielu posiadaczy pakietów graficznych na małe Atari, ta kich jak „Coala Micro Ilustrator" nerwowo przełyka ślinę, gdy widzi skromnego „Delux Paint'a". Połączenie, przez zdolnego programistę, dźwięku, grafiki i animacji w grę lub program demonstracyjny potrafi zwalić z nóg nawet posiadaczy Atari ST. W wielu grach kierujemy postaciami ze znanych kreskówek takich jak „Różowa Pantera" czy „Mickey Mouse" specjalizuje się w tym firma Byte Magic). Gry Dragon's Lair i Space Ace to w zasadzie filmy animowane, których jakości nie powstydziłby się nawet Disney, a Walker Demo to krótki film przypominający sceny z „Imperium kontratakuje". Niestety, trzy ostatnie dostępne są tylko dla szczęśliwych posiadaczy co najmniej 1MB RAM.

M.B.

GRAFIKA W AZTEC C

Wielu posiadaczy komputerów lubi bawić się grafiką. Nie da się ukryć, że połączenie Amigi i dobrego kompilatora może dać interesujące efekty. Jednak, zanim uda nam się otrzymać upragnioną kreskę od punktu 0,0 do 100,100 w kolorze zielonym, musimy pokonać wiele barier, jakie stworzyli programiści ROM-u Amigi. Postanowiłem pokonać te bariery i narysować kreskę na ekranie mojego monitora.

Najbardziej ucierpiał przy tym kot, który nie dość, że nie dostał obiadu, dopóki nie skończyłem, to jeszcze trafiłem go kapciem, gdy zbyt namolnie domagał się swoich praw.

Aby otrzymać wyżej wymienioną kreskę, należy najpierw napisać program rysujący ją, musimy zatem wybrać język, w jakim go napiszemy. Oczywiście najlepiej napisać program w C, gdyż nie ulega wątpliwości, że szybkość rysowania kreski w C jest zdecydowanie większa niż, dajmy na to, w Basicu.

Pierwszym krokiem w kierunku naszej kreski jest zorganizowanie danych ekranu w strukturę zwaną View. Struktura ta składa się z jednego lub kilku View-Port'ów, które określają części ekranu, oddzielone co najmniej jedną poziomą, pustą linią. Dzięki temu można otrzymać obraz, który składał się będzie z kilku części o różnych rozdzielczościach.

Aby opisać View Port, należy ustawić jego parametry: wysokość (w liniach), szerokość (w pikselach) i tryb graficzny, oraz podać wskaźnik do palety kolorów, informacji o pamięci ekranu (RasInfo) i następnego ViewPortu (jeśli takowy istnieje). Wysokość opisuje zmienna DHeight, szerokość DWidth, tryb graficzny zmienna Mode, która składa się z następujących bitów: DUALPF, PFBA, HIRES, LACE, HAM i SPRITES.

DUALPF włącza tryb dwuplanowy, tzn. obraz jest tworzony na dwóch ekranach nałożonych na siebie, gdy ustawiony jest bit PFBA, wtedy ekran drugi jest wyświetlany nad pierwszym.

HIRES ustawia tryb wysokiej rozdzielczości tj. 640 pikseli zamiast 320 (poziomo).

LACE ustawia tryb dzielony (ang.: „interlaced") tj. 400 zamiast 200 linii.

HAM włącza tryb Hold-And-Modify, który pozwala na otrzymanie wszystkich 4096 barw na jednym ekranie.

SPRITES informuje system, że używać będziemy sprite'ów.

Obraz tworzony w pamięci może być większy niż obszar objęty ekranem. Obraz taki, zwany rastrem, może mieć maksymalnie 1024x1024 punkty, a ponieważ nie mieści się w całości na ekranie, należy zdeklarować, który kawałek ma być wyświetlony. Określają go zmienne RHeight, RWidth, RyOffset, RxOffset, DHeight, DWidth, DyOffset, DxOffset. RHeight i RWidth opisują wysokość i szerokość rastru, RyOffset i RxOffset pozycję lewego górnego rogu okna, przez które oglądamy rysunek, DWidth i DHeight wysokość i szerokość tego okna, a DyOffset i DxOffset pozycję okna na monitorze komputera.

Zatem, aby otrzymać ekran, na którym można coś narysować należy:
— zdefiniować struktury, w których mają być zawarte dane obrazu
— otworzyć bibliotekę graficzną
— przygotować struktury View i ViewPort, a w niej: BitMap, RasInfo i mapę kolorów
— dołączyć strukturę View do listy rozkazów procesora graficznego.

Aby wyświetlić obraz zawarty w View, należy załadować go instrukcją LoadView() włączając tym samym DMA (direct memory access). Jak osiągnąć to w praktyce pokazuje program obok.

Na podstawie ROM Kernel
opracował
Marcin Bójko.

```
#include <exec/types.h>
#include <graphics/gfx.h>
#include <graphics/gfxbase.h>
#include <hardware/dmabits.h>
#include <hardware/custom.h>
#include <graphics/gfxmacros.h>
#include <graphics/rastport.h>
#include <graphics/view.h>
#include <exec/exec.h>

#define DEPTH 3
#define HEIGHT 201
#define WIDTH 640
#define BRAK_RAM -1000
#define X0 320
#define Y0 100

struct View v;              /* inicjalizacja struktur */
struct ViewPort vp
struct RasInfo ri;
struct BitMap b;
struct RastPort rp;
struct View *oldview;       /*  wskaźnik do starej struktury View */
struct ColorMap *cm;        /*  wskaźnik do tabeli kolorów cm */

short i,j,k,n;
struct ColorMap *GetColorMap();
struct GfxBase *GfxBase;    /* wskaźnik do biblioteki graficznej */

int MathTransBase,MathBase;

USHORT colortable[]={                        /* tabela kolorów */
 0x000,0xf00,0x0f0,0x00f,0x495,0x62a,0xf9c
};
UWORD *colorpalette;
main()
{
/* otwórz niezbędną bibliotekę graficzną */
 GfxBase=(struct GfxBase *)OpenLibrary("graphics.library",0);
 if(GfxBase == NULL)exit(1);
/* otwórz przydatne biblioteki graficzne: podstawową z ROM */
 if((MathBase=OpenLibrary("mathffp.library",0))<1)exit(-1);
/* i dodatkową z dysku (sin,cos itp) */
 if((MathTransBase=OpenLibrary("mathtrans.library",0))<1)exit(-1);

 InitView(&v);                  /* inicjalizuj View */
 v.ViewPort=&vp;               /* przyłącz ViewPort do View */
 InitVPort(&vp);                    /* inicjalizuj ViewPort */

 vp.DWidth=WIDTH;     /* ustaw szerokość i */
 vp.DHeight=HEIGHT;   /* wysokość ViewPort'u */
 vp.RasInfo=&ri;      /* informacje o rastrze znajdują się w ri */
 vp.Modes=HIRES;      /* wysoka rozdzielczość */

 InitBitMap(&b,DEPTH,WIDTH,HEIGHT); /* inicjalizuj mapę bitową */

 ri.BitMap=&b;                /* mapa bitowa znajduje się w b */
 ri.RxOffset=0;
 ri.RyOffset=0;

cm=GetColorMap(12);
colorpalette=cm->ColorTable;
for(i=0;i<8;i++)
{
 [illegible]
[illegible]

vp.ColorMap=cm;
for(i=0;i<DEPTH;i++)                          [illegible] */
                                             [illegible] */
 b.Planes[i]=[illegible]                      [illegible] */
 if(b.Planes[i]==NULL)exit(BRAK_RAM);

InitRastPort(&rp);           /* [illegible] */
rp.BitMap=&b;
MakeVPort(&v,&vp);
MrgCop(&v);                  /* [illegible]
                                [illegible] */
```



Przykład grafiki na „Amidze"

```
 SetRast(&rp,0);
 oldview=GfxBase->ActiView;    /* zapisz adres aktualnej struktury View w
                                  zmiennej oldview */

 LoadView(&v);                 /* załaduj nową strukturę View z v */

 SetAPen(&rp,1);               /* ustaw kolor pióra rysującego */
 Rysunek();                    /* narysuj rysunek */
 Delay(500);                /* odczekaj 10 sekund */

 LoadView(oldview);         /* załaduj starą strukturę */
 WaitTOF();                 /* poczekaj na koniec starej ramki */
 FreeMemory();              /* zwolnij pamięć */
 CloseLibrary(GfxBase);  /* pozamykaj wszystkie biblioteki */
 CloseLibrary(MathTransBase);
 CloseLibrary(MathBase);
}

FreeMemory()
{
 for(i=0;i<DEPTH;i++)                   /* zwolnij obszar który służył */
 {                                      /* pamięć ekranu */
  FreeRaster(b.Planes[i],WIDTH,HEIGHT);
 }
 FreeColorMap(cm);                      /* zwolnij obszar z tablicą kolorów */
 /* wymaż dynamiczne struktury pomocnicze */
 FreeVPortCopLists(&vp);
 FreeCprList(v.LOFCprList);
 return(0);
}

/* rysunek - to mała dusza - można tworzyć wiele ciekawych rysunków
 * polecam "Wstęp do grafiki komputerowej" J.D. Angela, poniższy przykład
 * został zaczerpnięty z tej właśnie książki */
Rysunek()
{
#include <math.h>
#include <libraries/mathffp.h>

 Spirala(0);
 Spirala(PI);
 Spirala(PI/2);
 Spirala(3*PI/2);
 Circle(320,100,160,90);
}
Circle(x,y,r1,r2)
int x,y,r1,r2;
{
 float a;
 Move(&rp,x,y+r2);
 for(a=0;a<TWO_PI;a=a+0.1)
 {
  Draw(&rp,(int)(r1*SPSin(a)+x),(int)(r2*SPCos(a)+y));
 }
}
Spirala(a0)
float a0;
{
 #define R 160
 #define N 3

 int x,y,i=0;
 float a,k;
 k=0.1*R/(N*TWO_PI);
 Move(&rp,X0,Y0);
 for(a=0;a<N*TWO_PI;a=a+0.1)
 {
  x=(int)(X0+k*i*SPSin(a+a0));
  y=(int)(Y0+k*i*SPCos(a+a0)/2);
  Draw(&rp,x,y);
  ++i;
 }
}
/* kompilowanie:   cc -s +l nazwa.c (kompilator Aztec C)
   konsolidowanie: ln nazwa.o -lc32 -lm32 */
```

ZABIĆ wirusa!

Pentagon wydał tysiące dolarów na antyseptykę po czym stwierdził, że jedynym sposobem na ustrzeżenie swoich komputerów przed wirusami jest odłączenie ich od sieci krajowej.

Wirusy atakują prawie wszystkie typy komputerów. Pojawiły się nawet na Commodore 64. Amigi „chorują" na nie nagminnie, stąd pomysł cyklu artykułów, w którym opisywane będzie działanie i sposób na niszczenie najgroźniejszych.

LAMMER EXTERMINATOR

Tak jak dawnymi czasy „Byte Bandit" siał popłoch na warszawskiej giełdzie, tak teraz niejaki „Lamer Exterminator" w Krakowie.

Siedział by on sobie w tym historycznym mieście, lecz niedawny zlot wywiózł go poza granice szanownego grodu... Jest to wybitnie złośliwy wirus. Nie tylko zapisuje się na boot-bloku, ale także niszczy jeden, wybrany losowo (a raczej pseudolosowo, bo na liczby losowe też jest algorytm, jak mawia BrOmba — kolega z klanu Spectrum), sektor na dysku. Zapisuje cały sektor słowem Lammer, przy czym nie liczy sumy kontrolnej bloku. Trafiony program na pewno nie będzie działał — próba wgrania go kończy się komunikatem systemowym: Volume „taki to a taki" has read write error — no i teraz niezależnie jak będziemy klęli — program jest nie do uratowania. Lammer z reguły atakuje najpotrzebniejszy lub najciekawszy program z dysku (pewnie zgodnie z jakimś prawem Murphy'ego. Tak się składa, że zanim skojarzymy nasze „padające" jak muchy dyski z robotą wirusa, a nie z niesolidnością firm Fuji i Kodak, jak się na początku zdawało, Lammer jest wszędzie, albo prawie wszędzie. Usunięcie go, dopóki rezyduje w pamięci, nie jest takie proste, gdyż przejmuje on kontrolę nad niektórymi przerwaniami systemowymi i niweluje wszystkie ataki — nie pozwala na zapisanie boot-bloku, a ponadto Virus X 2.0 nie wykrywa go, mimo, że ma Lammera w swoim spisie. Co prawda niektóre virus-killery zabijają go w pamięci, ale jest to pyrrusowe zwycięstwo, gdyż zaraz po zgłoszeniu triumfalnego komunikatu: „Lammer Eksterminator killed" pojawia się następny „Software failiture ..." i zabawa zaczyna się od początku. Jest to zapewne sprawka złośliwych programistów, którzy wypuścili drugą wersję wirusa, gdy pojawiły się przeciwciała zabijające pierwszą. Są jednak dwa sposoby na pozbycie się wrednego faga. Pierwszym jest wgranie „Virus Eksperta", który jako jedyny wygrywa pojedynek z Lammerem w RAM. Drugi sposób (zupełnie niesportowy) to wystartować komputer ze zdrowego dysku (czasem ciężko jest znaleźć takowy w swoich zbiorach), wgrać zwykłego virus-killera i wytłuc wszystkie okazy Lammera na dyskach. Teraz jeszcze tylko usunąć spustoszenia w zbiorach i pojedynek z Lammer Exterminator'em mamy za sobą. Ciekawe, z kim będzie następny?

wirusa zbadali i opisali
Marcin Bojko
i Mateusz Krauze

W stronę CHIWRITERa

Zdecydowanym standartem naszych biur, urzędów, instytucji i fabryk stały się komputery klasy IBM PC. Dostępne na ten sprzęt oprogramowanie wyznacza także pewne wzorce dla wielu programów. W dziedzinie języków programowania takim standartem jest kompilator Turbo Pascala firmy Borland. Nakładki systemowe reprezentowane są przez XTREE, Norton Commander i 1DIRPLUS. Wzorcem edytorów dyskowych jest pakiet Norton Utilities Advanced Edition. Oferowanych jest także wiele programów do edycji tekstów, na tle których wyróżnia się, pracujący w trybie graficznym, CHIWRITER.

```
HELP FILE.                    HIT PG UP, PG DN TO SCROLL ,ANY OTHER KEY TO CONTINUE
                                   Fonts
F1: Roman    ABC abc 123              F2: Symbol   ¶Łæ "¿° ┤┐+
F3: Italic   ABC abc 123              F4: Bold     ABC abc 123
F5: Gothic   ABC abc                  F6: Small    ABC abc 123
F7: Greek    ΓΔΩ γδω                  F8: Script   ĄBĆ ąbć
F9: Math I   ∀ ℵ ≤ □ ≅ ∩              F10:Math II  ABC ∫ √‾‾‾
```

Rys. 1. Fragment pliku Help. chi wyświetlany na ekranie komputera.

Edytor ten ma trzy podstawowe zalety:

1. Na ekranie widzimy tekst w takiej formie, w jakiej będzie na drukarce, tzn. różne alfabety, pochyłe, małe i pogrubione litery itd. (rys. 1.)
2. Bardzo łatwo można zdefiniować, bez żadnych przeróbek komputera i drukarki, własną czcionkę, np. polskie litery.
3. Możliwe jest przedstawienie w tekście skomplikowanych, wielopoziomowych wzorów matematycznych i struktur związków chemicznych.

Dodatkową zaletą nowszych wersji CHIWRITERa jest łączenie grafiki z tekstem.

Czy tylko na IBM PC?

O ile w miejscu pracy mamy najczęściej do dyspozycji IBM'a, to wracając do domu stajemy przed zupełnie innym sprzętem. Nasze domowe Atari, Spectrum, czy Commodore niespecjalnie pozwalają kultywować nawyki wyniesione z pracy. Diametralnie inny system operacyjny, oprogramowanie, a często brak monitora, stacji dysków i drukarki są przyczyną naszych frustracji.

Jeśli nie stać nas na sprzęt klasy IBM PC lub Atari ST, to czy możemy myśleć o unifikacji oprogramowania, korzystając z tańszego 8-bitowego komputera? Chcę rozważyć ten problem na przykładzie programu CHIWRITER.

Jakie warunki musi spełnić sprzęt, aby można spróbować rozwiązać to zagadnienie? Po pierwsze nasz komputer domowy musi umożliwiać wymianę informacji z maszyną w pracy. Może się to odbywać przy pomocy dyskietki lub złącza RS 232C i ewentualnie modemu. Po drugie powinniśmy mieć możliwość wyświetlania na ekranie grafiki o rozdzielczości 640*200 punktów (standart CGA na IBM PC). Wymagania te w łatwy sposób spełniają dostępne na naszym rynku komputery standartu CP/M: Amstrady PCW 8256/ /8512, CPC 6128 i Commodore 128 D. Nie oznacza to, że właściciele innych popularnych urządzeń stoją na straconej pozycji, ale w ich przypadku, z powodu ograniczeń sprzętowych, zagadnienie implementacji CHIWRITERa jest trudniejsze.

Generatory znaków programu CHIWRITER

Przejdźmy teraz do problemów programistycznych. CHIWRITER pracując w trybie graficznym, korzysta z wielu własnych generatorów znaków, zarówno ekranowych, jak i drukarkowych. W przypadku ekranu matryca znaku ma rozmiary 10*8 punktów i każdy znak pamiętany jest w kolejnych 10 bajtach (rys. 2a). Dla drukarki mamy dwa rodzaje matryc — tryby draft i NLQ. Matryca trybu draft ma rozmiar 10*12 punktów i przechowywana jest w pamięci komputera jako ciąg kolejnych 20 bajtów (rys. 2b). Analogicznie w trybie NLQ matryca ma rozmiary 20*12 i wymaga 40 bajtów.

Kolejne znaki przechowywane są w plikach generatorów znaków. W tabelce 1 podano listę typowych generatorów ekranowych programu CHIWRITER. Te same generatory dla drukarki posiadają rozszerzenia DFT (tryb

```
                 K o l u m n y

                1 2 3 4 5 6 7 8

bajt 0          0 0 0 0 0 0 0 0       wiersz  1
bajt 1          0 1 1 1 1 1 1 0       wiersz  2
bajt 2          0 1 1 0 0 1 1 0       wiersz  3
bajt 3          0 1 1 0 0 1 1 0       wiersz  4
bajt 4          0 1 1 0 0 1 1 0       wiersz  5
bajt 5          0 1 1 1 1 1 1 0       wiersz  6
bajt 6          0 1 1 0 0 1 1 0       wiersz  7
bajt 7          0 1 1 0 0 1 1 0       wiersz  8
bajt 8          0 1 1 0 0 1 1 0       wiersz  9
bajt 9          0 0 0 0 0 0 0 0       wiersz 10

                7 6 5 4 3 2 1 0

                     B I T Y
```

Rys, 2a, Matryca ekranowego (,SFT) generatora znaków programu CHIWRITER

```
                     K o l u m n y
                                   1
                1 2 3 4 5 6 7 8   9 0 1 2

bajt  0         0 0 0 0 0 0 0 0   0 0 0 0 x x x x     wiersz  1
bajt  2         0 0 1 1 1 1 1 1   1 1 0 0 x x x x     wiersz  2
bajt  4         0 0 1 1 0 0 0 0   1 1 0 0 x x x x     wiersz  3
bajt  6         0 0 1 1 0 0 0 0   1 1 0 0 x x x x     wiersz  4
bajt  8         0 0 1 1 0 0 0 0   1 1 0 0 x x x x     wiersz  5
bajt 10         0 0 1 1 1 1 1 1   1 1 0 0 x x x x     wiersz  6
bajt 12         0 0 1 1 0 0 0 0   1 1 0 0 x x x x     wiersz  7
bajt 14         0 0 1 1 0 0 0 0   1 1 0 0 x x x x     wiersz  8
bajt 16         0 0 1 1 0 0 0 0   1 1 0 0 x x x x     wiersz  9
bajt 18         0 0 0 0 0 0 0 0   0 0 0 0 x x x x     wiersz 10

                7 6 5 4 3 2 1 0   7 6 5 4 3 2 1 0

                          B I T Y
```

Rys, 2b, Matryca drukarkowego (,DFT) generatora znaków programu CHIWRITER

```
0000   10 62 6F 6C  64 2E 53 46  54 00 00 00  00 00 BA 06  ,bold,SFT.....:.
0010   5E 01 00 08  0A 00 08 00  05 00 00 00  00 00 00 00  ^...............
```

Rys, 3a, Nagłówek pliku generatora znaków programu CHIWRITER

```
0158   00 38 38 38  38 38 00 38  00 00 00 EE  EE EE 00 00  ,88888,8,,,nnn,,
0168   00 00 00 00  00 6C FE FE  6C FE FE 6C  00 00 00 38  .....l~~l~~l...8
0178   7E F0 7C 1E  FC 38 00 00  00 E6 EE 1C  38 70 EE CE  ~pl.l8...fn.8pnN
0188   00 00 00 3C  76 3C 78 FB  EE 38 00 00  00 70 70 E0  ...<v<x{n;...pp`
0198   00 00 00 00  00 00 00 1C  38 70 70 70  38 1C 00 00  ........8ppp8...
01A8   00 70 38 1C  1C 1C 38 70  00 00 00 10  FE 7C 7C FE  .p8...8p....~||~
01B8   10 00 00 00  00 00 00 38  38 FE 38 38  00 00 00 00  .......88~88....
01C8   00 00 00 00  30 30 60 00  00 00 00 00  FC FC 00 00  ....00`.....||..
```

Rys, 3b, Bajty pierwszych znaków generatora,

```
BOLD      SFT
GOTHIC    SFT
GREEK     SFT
ITALIC    SFT
MATHI     SFT
MATHII    SFT
PLBOLD    SFT
PLITALIC  SFT
PLSMALL   SFT
POLISH    SFT
ROMAN     SFT
SCRIPT    SFT
SMALL     SFT
SPECIAL   SFT
SYMBOL    SFT
SYSTEM    SFT
```

TAB. 1. Lista plików ekranowych generatorów znaków programu CHIWRITER

```
B[0],.B[11] - nazwa pliku generatora

B[16] - ilość znaków w generatorze

B[17] - kod ASCII - 32 dla pierwszego znaku

B[19] - ilość kolumn w znaku

B[20] - ilość wierszy w znaku

B[344] - pierwszy bajt definiujący pierwszy znak
```

TAB 2. Opis niektórych pól generatora znaków programu CHIWRITER

```
\3Move the cursor to the next \4*\3, turn insert mode off, add about\. 10
\+                                                  \05______6
\+                                                    \62
\+                                          \1-b \9+ \0r \1b -4ac
\+
\3levels and copy this formula: \1x\.\.\. = \2_____________\3. When you\. are \!
\-                                    \61,2
\-                                                \12a
\-
\-
\-
\3done, turn insert mode back on! You will find the \2_ \3in font 2, the \"
small \62 \3in font 6, the \9+ \3in font 9 and the characters to make\. the
```

Rys. 4a. Fragment pliku TUTORIAL.CHI (ASCII).

*Move the cursor to the next *, turn insert mode off, add about 10*

levels and copy this formula: $x_{1,2} = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$*. When you are*

done, turn insert mode back on! You will find the – in font 2, the small ± in font 6, the ± in font 9 and the characters to make the

Rys. 4b. Ten sam fragment interpretowany na ekranie Joyce'a

draft) i PFT (tryb NLQ). Organizacja plików generatorów przedstawiona jest na rysunkach 3a i 3b. W nagłówku zbioru znajduje się szereg istotnych parametrów generatora, niektóre z nich zostały zebrane w tabelce 2. Najważniejsza dla nas informacja zaczyna się od adresu 344, zawierającego pierwszy bajt matrycy pierwszego znaku.

Jeśli dalej myślimy o programie CHIWRITER na naszym komputerze, to wszystkie te generatory, a jest ich dużo, musimy przenieść na nasz sprzęt. Najprościej zrobić to przy pomocy stacji dysków 5 i 1/4 cala, jeśli nasz komputer umie czytać dyskietki IBM PC. Alternatywnym, mniej wygodnym rozwiązaniem jest użycie złącza RS 232 C i oprogramowania komunikacyjnego. Właścicielom komputerów CP/M'u istotnie pomoże program COMHEX, opisany w „Bajtku" 6/88. Bezpośrednie zastosowanie komend COPY i PIP nie jest możliwe, ponieważ transferowane pliki są binarne, tzn. wykorzystują 8 bitów każdego słowa, a transmisja jest 7-bitowa.

Rysowanie znaku na ekranie

Gdy uporamy się z przeniesieniem generatorów, co może zająć trochę czasu, stajemy przed kolejnym problemem. Musimy wprowadzić plik generatora do pamięci komputera i napisać program, który będzie umiał rysować znaki na ekranie wg zadanych matryc. Ponieważ typowe matryce większości komputerów domowych mają rozmiary 8*8, nie możemy podmienić generatora systemowego generatorem CHIWRITERA. Potrzebna jest procedura PLOT (X, Y) zapalająca na ekranie punkt o współrzędnych X, Y. Dla Amstrada CPC 6128 pascalowa procedura PLOT znajduje się na listingu 3. Wersję dla Commodore 128D można znaleźć w numerach „Komputera" z 1988 roku. W przypadku Amstrada PCW, można posłużyć się pakietem graficznym opisanym w 7,8,9 i 12 numerze „Bajtka" z 1988 roku. Na listingu 3 przedstawiono także, niezależną od komputera, procedurę DC, która rysuje na ekranie znak c w punkcie x, y.

Co jeszcze?

Zatrzymajmy się na chwilę w naszej ciężkiej pracy i zastanówmy się, do czego doszliśmy i co nam jeszcze pozostało. Mamy przeniesione generatory znaków i umiemy rysować dowolne litery, alfabety i czcionki na ekranie naszego komputera. Następne cele, w porządku rosnącym ze względu na stopień trudności, możemy sformułować następująco:

1. Napisanie programu, który będzie wyświetlał zbiór przygotowany przez CHIWRITERA na ekranie komputera.
2. Napisanie programu, który wydrukuje taki zbiór na drukarce.
3. Napisanie właściwego edytora.

Niestety z dwóch powodów — mało miejsca w „Bajtku" i niskie honoraria autorskie — ograniczę się tylko do opisania pierwszego problemu i krótkich szkiców dla dwóch następnych zagadnień.

Interpretacja zbiorów CHIWRITERa

Program CHIWRITER, ze względu na swe bogate możliwości, zapamiętuje edytowany tekst w pewien specjalny sposób (rys. 4a). Do zmiany czcionki, do zaznaczenia miękkich spacji, różnych odstępów między wierszami i innych cech tekstu służą sekwencje sterujące. Przykładowo zmiana generatora odbywa się przy pomocy ciągu dwóch znaków '/n', gdzie n jest numerem generatora. Miękka spacja zaznaczana jest w zbiorze przez '/.'. Ogólnie biorąc specjalne sekwencje sterujące są dwuznakowe i zaczynają się od znaku '/', o kodzie ASCII równym 92. Przyznaję, że nie udało mi się rozszyfrować znaczenia wszystkich sekwencji sterujących, ale te znalezione pozwalają na poprawną interpretację zbiorów HELP.CHI i TUTORIAL.CHI, także przeniesionych z IBM PC.

Na listingu 1 w programie ChiToScreen znajduje się procedura DisplayFile, która interpretuje zbiór przygotowany przy pomocy CHIWRITERa. Korzysta ona z wcześniej opisanej procedury DC (x, y, ch), wyświetlającej na ekranie w punkcie x, y znak ch. Znajdująca się w programie procedura NewLine steruje przejściem do nowego wiersza tekstu. Bardzo ważną rolę odgrywa procedura LoadSgen, ładująca do pamięci komputera zmodyfikowany, ekranowy generator znaków CHIWRITERa. Zmiana generatora polegała na obcięciu pierwszych 344 bajtów w oryginalnym generatorze. Pozwoliło to na jego zmiejszenie do rozmiaru 1KB.

Amstrad PCW 8256/8512

Program ChiToScreen przedstawiono na listingu 1 w wersji na komputer Amstrad PCW 8256. Procedura DC, odwołująca się do RSX'a uzyskanego z asemblerowego programu DC.MAC, przedstawionego na listingu 2, pozwala na 3-krotne przyśpieszenie wyświetlania zbioru, w stosunku do wersji posługującej się procedurą PLOT. Przygotowanie programu DCH.COM ze zbiorów DCH.PAS (listing 1) i DC.MAC (listing 2) wymaga następujących operacji: (patrz np. „Bajtek" 5/88 „Własne znaki na ekranie Joyce'a")

1. Kompilacja na dysk Turbo Pascalem, ze zmniejszonym adresem końcowym (ang. END ADDRESS), zbioru DCH.PAS.
2. Przygotowanie RSX'a ze zbioru DC.MAC:

```
M80=DC.MAC
LINK DC [OP]
REN DC.RSX=DC.PRL
```

3. Połączenie programu z RSX'em:

```
GENCOM DCH.COM DC.RSX
```

Amstrad CPC 6128 i Commodore 128D

W przypadku CPC 6128 sytuacja jest prostsza, ponieważ nie trzeba tworzyć RSX'a. Ze zbioru DCH.PAS musimy wyrzucić procedurę DC i wstawić w to miejsce procedury PLOT i DC znajdujące się na listingu 3. Dla Commodore 128D należy zmienić procedurę PLOT.

Otrzymany program DCH.COM uruchomiłem na Amstradzie PCW 8256. Na rysunku 4b znajduje się fragment zbioru TUTORIAL.CHI zinterpretowany i wyświetlony na ekranie Joyce'a, przy pomocy opisanego programu.

Drukowanie zbiorów CHIWRITERa na drukarce wymaga pewnej modyfikacji programu DCH.

PAS. Należy uwzględnić inny rozmiar matrycy znaku i napisać procedurę DC w ten sposób, aby wysyłała znak na drukarkę, posługując się trybem graficznym. Przykład zastosowania trybu graficznego drukarki przedstawiono w „Bajtku" 1/88 (program BitImage).

Napisanie kompletnego edytora jest największym problemem ale jest to już w zasadzie sprawa dobrego rzemiosła programistycznego, które może być wsparte odpowiednią literaturą i przykładami. Firma Borland oferuje między innymi pakiet procedur do Turbo Pascala, pozwalających na pisanie edytorów. Pewne informacje można znaleźć w innym produkcie tej firmy, jakim jest Toolbox do baz danych, którego wersja pod CP/M'em, jest również dostępna i pozwala tworzyć naprawdę bardzo dobre bazy danych w tym systemie operacyjnym.

Po co to wszystko?

Podając informacje o organizacji plików CHIWRITERa i jego generatorów chciałem pokazać, że możliwe jest napisanie takiego programu na komputer inny niż IBM PC.

Niestety tragiczny stan w dziedzinie ochrony praw autorskich w Polsce jest przyczyną, że tego nie zrobię. Myślę, że taka sytuacja jest niezwykle stresująca dla wielu naszych programistów, którzy praktycznie zrezygnowali z tego rodzaju twórczości. Można pisać specjalistyczne programy na zamówienie bogatych sponsorów, ale trudno napisać program powszechnego użytku, zwłaszcza na popularny sprzęt domowy, gdy wiadomo, że sprzedamy jeden egzemplarz, bo handlem następnymi zajmą się złodzieje z giełdy.

Brak jakichkolwiek uregulowań prawnych w tej dziedzinie, brak hamulców moralnych przed kradzieżą czyjegoś dorobku intelektualnego owocuje sytuacją, w której tracimy wszyscy.

Jonasz Mayer

■

```
(************************************************************)
{        Wyswietlanie na ekranie Joyce'a                       }
{         zbiorow programu CHIWRITER                           }
{                                                              }
{              Plik DCH.PAS                                    }
{               wersja 1.0                                     }
{            (C) JM Lipiec 1989                                }
{                                                              }
{ Uwagi:                                                       }
{                                                              }
{ 1. program korzysta ze zmodyfikowanych generatorow znakow}
{    programu CHIWRITER, przeniesionych z komputera IBM PC.}
{ 2. Celem przyspieszenia dzialanie program odwoluje sie   }
{    do RSX numer 75, ktory wyswietla znaki na ekranie wg  }
{    podanego generatora znakow. RSX generowany ze zbioru  }
{    DC.MAC musi byc zainstalowany przed wykonaniem        }
{    programu pascalowego albo dolaczony do niego przy     }
{    pomocy polecenia GENCOM DCH.COM DC.RSX                }
{                                                              }
(************************************************************)

const
  xScrMax = 719;    { rozmiary ekranu Joyce'a }
  yScrMax = 255;

  MaxRow = 9;   { MaxRow+1 - liczba wierszy w matrycy znaku}
  MaxCol = 7;   { MaxCol+1 - liczba kolumn w matrycy znaku }

  NoGen     = 10;       { liczba generatorow znakow }
  GenSize   = 100;      { ilosc znakow w generatorze }
  ext       = '.SFT';   { domyslne rozszerzenie zbioru }

  GFName : array [1..NoGen] of String[10] =
     ('ROMAN','SYMBOL','ITALIC','BOLD', 'GOTHIC',
      'SMALL','GREEK', 'POLISH','MATHI','MATHII');

  half : boolean = false;

type
  schar    = array (.0..MaxRow.)  of byte;           { znak }
  sgen     = array (.0..GenSize.) of schar;  { gen. znakow }
  Str20    = string [20];

var
  f : text;
  name : str20;
  GenBuffer : array [1..NoGen] of sgen;
  x,y   : integer;                     { Poz. drukowania }
  dy    : integer;              { akt  odstep m/wierszami }
  Cfont : byte;                            { Akt. no. fontu }

Procedure LoadSgen (no:byte; filename:str20);
(************************************************************)
{ Procedura laduje wskazany generator znakow ze zbioru       }
{ dyskowego filename.                                         }
(************************************************************)
var
    gfile : file;
begin
  assign (gfile,filename+ext);     (*$I-*)
  reset  (gfile);                  (*$I+*)
  if IOresult<>0
  then begin
         writeln('Brak generatora ',filename,'. STOP');
         Halt;
       end
  else begin
         BlockRead (gfile,genBuffer[no],8);
         Writeln (filename:8,ext,' - font ',no:1);
       end;
  close(gfile);
end;  (* load pgen *)

procedure NewLine;
(************************************************************)
{ Procedura powoduje przejscie do nowej linii.                }
(************************************************************)
var ch : char;
begin
  x := 0;     y := y + dy;
  if half
  then begin y := y-dy+5;  half := false;
       end;
  if y>yScrMax-18
  then begin
         y := 5;
         Gotoxy (1,32);
         ClrEol;
         write(' nacisnij dowolny klawisz ');
         repeat until keypressed; read (kbd,ch);
         if ch='x'
         then begin
                write (#27#49#27'e');
                Halt;
              end;
         clrscr;
      end;
end;  { of NewLine }

procedure DC (var x,y : integer; c : char);
(************************************************************)
{ Procedura drukuje  znak na ekranie wg generatora o         }
{ numerze cfont. Korzysta z RSX'a !                          }
(************************************************************)
var
   d : record
          xc   : integer;
          yc   : byte;
          adrT : integer;
       end;
begin
  d.xc := x;
  d.yc := y;
  d.adrT := addr (GenBuffer[cfont][ord(c)-32]);
  bdos (75,addr(d));
  x := x + MaxCol + 1;
  if x >xScrMax-MaxCol
  then Newline;
end; { of DC }

procedure BreakPage;
(************************************************************)
{ Procedura zaznacza koniec strony w zbiorze.                }
(************************************************************)
var i : integer;
begin
  for i := 0 to 79  do DC(x,y,'-');      NewLine;
end; { Break Page }

procedure LogIn;
(************************************************************)
{ Procedura wypisuje naglowek programu i czyta kolejne       }
{ generatory znakow.                                          }
(************************************************************)
var i : byte;
begin
  ClrScr;
  writeln ('*** CHIWRITER  PCW 8256  DEMO ***');
  writeln;
  x := 500;  y := 5;  cfont := 1;
  for i := 1 to 10
  do begin
       LoadSgen(i,gfname[i]);
       DC (x,y,chr(i+48));
     end;
  writeln;
end; { LogIn }

procedure GetFileName;
(************************************************************)
{ Procedura pobiera nazwe zbioru i otwiera go. Jesli nie     }
{ ma zbioru na dysku, program jest przerywany. Nie podanie   }
{ zadnej nazwy konczy program.                                }
(************************************************************)
begin
    write ('nazwa pliku: ');    readln (name);
    if name = ''
    then begin
           ClrScr;  { pusta nazwa konczy program }
           Halt;
         end;
    if pos ('.',name)=0
    then name := name + '.CHI';
    assign (f,name);    {$I-}
    reset(f);           {$I+}
    if IOresult<>0
    then begin
           writeln ('Brak zbioru !') ; Halt;
         end;
end; { GetFileName }

procedure DisplayFile;
(************************************************************)
{ Czyta zbior Chiwritera z dysku i wyswietla go na ekranie }
{ interpretujac znaki sterujace.                             }
(************************************************************)
var
      ch : char;
begin
 write (#27#48#27'f');     { status off, cursor off }
 ClrScr;
 x := 0;  y := 5;  dy := 10;  cfont := 1;

 while not eof(f)
 do begin

      while not eoln(f)
      do begin
           read(f,ch);
           if ch='\'
           then begin
                  read(f,ch);
                  case ch
{ zmiana }        of   '0' : cfont := 10;
{ fontu  }        '1'..'9' : cfont := ord(ch)-48;

{ dod. wiersz gorny }  '+' : half := true;
{ dod. wiersz dolny }  '-' : y  := y - dy + 5;

{ zmiana odstepu   }   '!' : dy := 10;
{ miedzy wierszami }   '"' : dy := 15;
                       '#' : dy := 20;

{ soft space )         ',' : x := x + MaxCol+1;
{ soft CR    }         ';' : ;

{   nieznane  }        '&' : ;
{   znaki     }        '''': ;
{.  sterujace }        '/' : ;

{ znak konca zbioru }  '=' : ;
                  else begin
                         gotoxy (1,32); ClrEol;
                   write ('Nieznany znak ster.: \',ch);
                       end;
                  end;              { of case }
                end
           else begin
                  if ch=#$0c then BreakPage;
                  if ch=' '  then x := x + MaxCol + 1;
                  if (ch>' ') and (ch<#126)
                  then DC (x,y,ch);
                end;
      end;                  { of eoln }

      readln(f);
      NewLine;
    end;                    { of eof }

    gotoxy (1,32);
    ClrEol;
    write ('Koniec pliku. Nacisnij dowolny klawisz.');
    repeat until keypressed;
    read (kbd,ch);
    ClrScr;
    write (#27#49#27'e');  { status on, cursor on }
end; { Display File }

begin  { MAIN }
  LogIn;
  repeat
    GetFileName;
    if name<>'' then DisplayFile;
  until name='';
end.

(************************************************************)

Listing 1. Plik DCH.PAS
```

```
; ***********************************************
; *               Plik DC.MAC                   *
; *                ver. 1.0                     *
; *          (C) JM  April 1989                 *
; *                                             *
; *  parametry:                                 *
; *   rejestr c = 75                            *
; *   rejestr de - adres bloku parametrów       *
; *   przekazywanych z programu użytkownika:    *
; *   nazwa   znaczenie                         *
; *     x       wspól. x znaku   (word)         *
; *     y       wspól. y znaku   (bajt)         *
; *     adrT   adres matrycy znaku w gen. (word)*
; ***********************************************
        .z80
        cseg
        ds   6
        jp   start
next:db   0C3H,0,0,0,0,0FFh,0
        db   'DrawChar',0,0,0
; ****************************************
start:
        ld   a,c
        cp   75
        jp   nz,next ;
begin:                  ; pobierz parametry
        ex   de,hl
        ld   e,(hl)
        inc  hl
        ld   d,(hl)   ; de := x
        inc  hl
        ld   c,(hl)
        ld   b,0
        push bc
        pop  iy        ; iy := y
        inc  hl
        ld   b,(hl)
        inc  HL
        ld   h,(hl)
        ld   l,b       ; hl := adrTab
; ****************************************
DC:  ld   b,10      ; 10 wierszy
L1:  push bc        ; zapamiętaj licznik wierszy
        push de
        ld   a,(hl)   ; a = bajt znaku z matrycy
        ld   b,8      ; 8 bitów (kolumn)
L2:  rlca           ; carry := bit (kolumna)
        jr   nc,NCY
        push bc        ; zapamiętaj licznik kolumn
        push af        ; zapamiętaj bajt matrycy
        push de        ; zapamiętaj x
        push hl        ; zapamiętaj adres matrycy
        push iy
        pop  hl        ; hl := y
        ld   bc,PLOT   ; *
        call 0FC5AH    ;  * wołaj proc. plot
        dw   00E9H     ; *
        pop  hl        ; odtwórz adres matrycy
        pop  de        ; odtwórz x
        pop  af        ; odtwórz bajt matrycy
        pop  bc        ; odtwórz licznik kolumn
NCY:  inc  de        ; x := x+1 (poz. plot'a)
        djnz L2        ; pętla do następnej kolumny
        inc  hl        ; następny wiersz matrycy
        inc  iy        ; y := y+1 (poz. plot'a)
        pop  de
        pop  bc        ; odtwórz licznik wierszy
        djnz L1        ; pętla do następnego wiersza
        ret

PLOT:                  ; (de=x,hl=y)
        add  hl,hl
        ld   bc,0b600h
        add  hl,bc
        ld   c,(hl)
        inc  hl
        ld   b,(hl)
        ld   a,c
        and  0f8h
        ld   l,a
        ld   h,b
        add  hl,hl
        add  hl,de
        ld   a,l
        and 0f8H
        ld   l,a
        ld   a,c
        and  7
        or   l
        ld   l,a
        ld   a,e
        and  7
        inc  a
        ld   b,a
        xor  a
        scf
LXXP;rra
        djnz LXXP
        or   (hl)
        ld   (hl),a
        ret
; *****************************************
        end
;
; Listing 2. Plik DC.MAC
;
```

```
procedure PLOT (x,y : integer);
{*****************************************}
{ procedura PLOT w wersji dla CPC 6128      }
{*****************************************}
begin
  InLine ($2A/x/$EB/$2A/y/$CD/$5A/$FC/$EA/$BB);
end; (* PLOT *)


procedure DC (var x,y : integer; c : char);
{*****************************************}
{ procedura drukujaca w trybie graficznym   }
{ znak na ekranie. Korzysta z procedury PLOT.}
{*****************************************}
type
  Matrix   = array [0..MaxRow,0..MaxCol] of byte;
var
    i,j, k : integer;
    M : Matrix;

  procedure S_to_M (var M : matrix; var s : schar);
  {*****************************************}
  { Zamienia MaxRow+1 bajtow znaku na matryce  }
  { znaku o rozmiarach (MaxRow+1)*(MaxCol+1).  }
  {*****************************************}
  var i,j,x,mask : byte;
  begin
     for i := 0 to MaxRow
     do begin
              x := s(.i.);   mask := $80;
              for j := 0 to MaxCol
              do begin
                      if (x and mask)<>0
                      then M(.i,j.) := 1
                      else m(.i,j.) := 0;
                      mask := mask shr 1;  (* div 2 *)
                 end;
        end;
  end; (* of S to M *)

begin { DC }
  S_to_M (M,GenBuffer[ord(c)-32]);
  for i := 0 to MaxRow
  do begin
       k := MaxRow - i;
       for j := 0 to MaxCol
       do if M[i,j]<>0
          then
                plot (x+j,y+k,1);  { right }
     end;
  x := x + 8;
end; { of DC }
```

Listing 3. Procedura DC w wersji korzystajacej z procedury PLOT (x,y).

Smok i codzienność

Przerażający wrzask przebiegł przez zamkowe komnaty. To smok skaleczony złamaną szczoteczką do zębów zawył głośno. Następnym jego zajęciem miało być szczotkowanie grzywy, lecz stało się to niemożliwe z powodu zagubienia grzebienia (grabi). Rozczochrany smok udał się do toalety. Tak również czekała go niespodzianka: pięknie wyrysowany grafik najpierw instruował, jak należy skorzystać z papieru toaletowego, lecz kończył się stwierdzeniem, że papieru tego brak.

Smok, który miał w zamiarze ożenek, zapłakał rzewnymi łzami.

— Jak mam się ożenić z Księżniczką Ośmiornicą — chlipnął smoczek — gdy nie posiadam nawet złamanego grzebienia, ani innych przedmiotów codziennego użytku. Księżniczka będzie bardzo wymagająca, a jej posag niewielki...

Czy nasze smoki płaczą podobnie?

Jeśli nie posiadacie programów użytkowych na stację dysków TIMEX, to wasz smok (stacja) właśnie tak płacze.

Co może osuszyć łzy smoka? Podaję niżej krótkie opisy programów dostępnych na giełdzie oraz tych, którymi dysponuje redakcja.

MEGA PHANTOM — profesjonalnie napisany program służący do transferu programów z dyskietki na taśmę. Program ten jest ewenementem na rynku oprogramowania, ponieważ oprócz tego, że jest bardzo dobrze opracowany, zawiera nawet winietę autorską. Posiadaczy interfejsu AY zaciekawią na pewno wspaniałe efekty dźwiękowe!

Panom Langerowi i Burczyńskiemu należą się brawa za ten produkt.

ZEBRA COPY — również profesjonalny program, o działaniu przeciwnym do poprzedniego. Umożliwia on przenoszenie programów z taśmy na dyskietkę. Wiele opcji; automatyczne poprawianie loaderów do współpracy z TOS-em, turbo taśmowe itp. znacznie ułatwiają pracę.

Autorem Zebry jest tajemniczy Mr KATO, autor wielu programów kopiujących, np. Turbo Compress Copy.

DEER COPY — program kopiujący dla pojedyńczego systemu dyskowego. Wygodny w obsłudze, z wykrywalnością błędów oraz wygodną selekcją plików do skopiowania. Jego poważną zaletą jest ilość wolnej pamięci pozwalająca na skopiowanie pliku o długości do 39730 bajtów.

TWOCOPY — tym razem program przenoszący pliki między dwoma napędami (z A na B lub odwrotnie). Wykonany estetycznie (okienka), zawierający wiele komunikatów informujących o stanie systemu, jest idealnym programem dla posiadaczy stacji dwukieszeniowych. Program przesyła pliki bezpośrednio między napędami, nie zajmując RAM-u komputera.

Autorem dwóch ostatnich programów jestem ja i są one dostępne w redakcji „Bajtka".

Wracając do ożenku smoka: do systemu TOS można podłączyć napęd 5.25 cala czterdziesto- lub osiemdziesięcióścieżkowy. Najlepiej jest powierzyć to doświadczonemu elektronikowi, który dokona w razie potrzeby odpowiednich przeróbek. Nie są one w ogóle potrzebne w przypadku napędu **TEAC FD 55B-01-U**, wystarczą kabelki wewnątrz obudowy stacji.

Gdy smok ożeni się z Ośmiornicą (80 ścieżek), system TOS pozwala na uzyskanie 620 KB na stronie dystetki HD (high density), lecz do normalnej składni komendy formatującej należy dodać literkę **d** (double) na końcu.

Smoku, ożeń się, żony ułatwiają życie!

Maciej Pietraś

PS. Proszę Czytelników o listy, zapytania dotyczące eksploatacji systemu TOS i stacji TIMEX.

WINNIE-THE-







Dom Kangurzycy




dół

drzewo

tunel








Dom Puchatka






dół

Dom Prosiaczka






drzewo

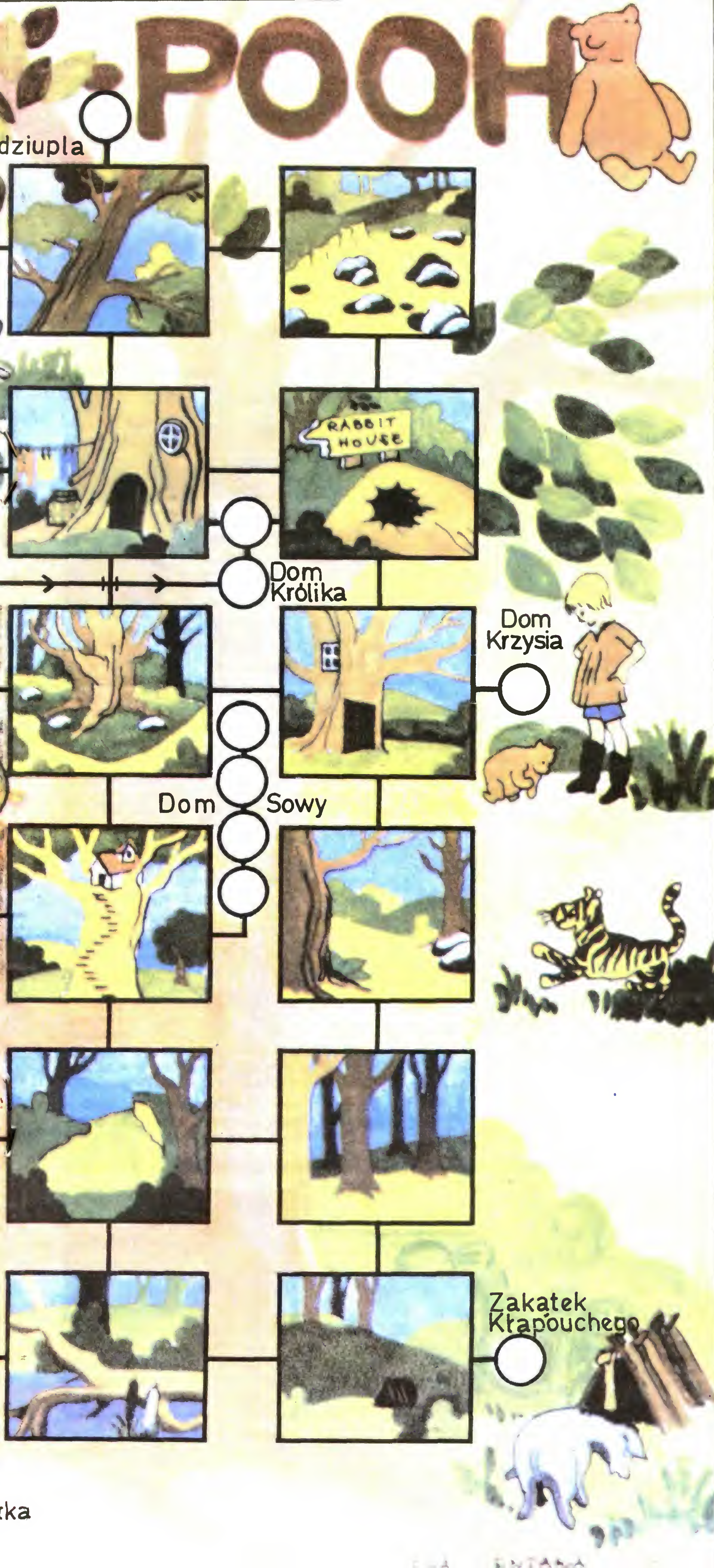




brzeg rzeczki

w kuferku jest dyskiet

Bajtek



KUBUŚ PUCHATEK

Winnie-the-Pooh, znany w Polsce jako Kubuś Puchatek jest bohaterem lubianych książeczek „Kubuś Puchatek" i „Chatka Puchatka". Jeśli czytaliście je, nie będziecie mieli trudności z grą.

Znajdziecie w niej ulubione postacie: Krzysia (Christopher Robin), Puchatka, Prosiaczka (Piglet), Kłapouchego (Eyeore), Królika (Rabbit), Kangurzycę (Kanga) z Maleństwem (Roo) oraz rozbrykanego Tygrysa (Tiger).

Zrywający się co chwila wiatr rozrzuca po Lesie różne przedmioty. Czy pamiętacie, jak strącił z drzewa dom Sowy? Byli wtedy u niej na podwieczorku Puchatek z Prosiaczkiem. To ten sam wiatr.

Porozrzucane przedmioty należy oddać ich właścicielom. Znając mieszkańców Lasu i ich obyczaje zrobicie to bez trudu. Wiecie, że Kłapouchy lubi oset (thistle), Puchatkowi bardzo potrzebny był balon (baloon), gdy wybierał się po miód, zaś przed lustrem (glass) co rano robi gimnastykę.

Po Lesie poruszacie się wybierając kierunek geograficzny. Gdy znajdziecie jakiś przedmiot, możecie podnieść go, ale tylko jeśli macie wolne ręce. W przeciwnym wypadku można trzymaną rzecz położyć, a podnieść znalezioną. Dzięki mapie możliwe będzie dotarcie do pozostawionych przedmiotów. Może również zerwać się wiatr i porozrzucać przedmioty, które przyjmą inne położenie.

Inne niespodzianki to Bardzo Rozbrykany Tygrysek, który przebiegając przez Las może wytrącić trzymany przedmiot z rąk. Gęsta mgła bardzo utrudnia poruszanie, ale od czego mapa!

Gra nie ma ograniczenia czasowego, dlatego warto pobyć dłużej w niektórych miejscach. Można śpiewać piosenki na Muzycznej Polanie lub szukać skarbów pod kamieniami. Można pójść na spacer w dół rzeki lub wspinać się po drzewach. Można tropić Łesice i Łysice (Woozle, Wizzle) lub zwiedzić domy przyjaciół.
Możliwości jest niezwykle dużo.

Jeśli oddacie już wszystkie przedmioty właścicielom i chcecie skończyć grę, idźcie na północ. Jest tam miejsce na piknik i czeka tam niespodzianka! Gry nie trzeba kończyć po gratulacjach, chodząc dalej po lesie znajdziecie następne przedmioty.

Polskie nazwy w opisie gry zaczerpnięte są z książek w przekładzie pani Ireny Tuwim.

Firma: Sierra On-Line
Komputer: Atari ST, Commodore 64

Ewa Zientara

10 BAJTKOWA LISTA PRZEBOJÓW (12/89)

Nowe pomysły, super strzelaniny, magalabiryntówki. Zalew nowości. Producenci strzelają grami jak z karabinu. My gramy jak opętani. W co gramy w tym miesiącu?

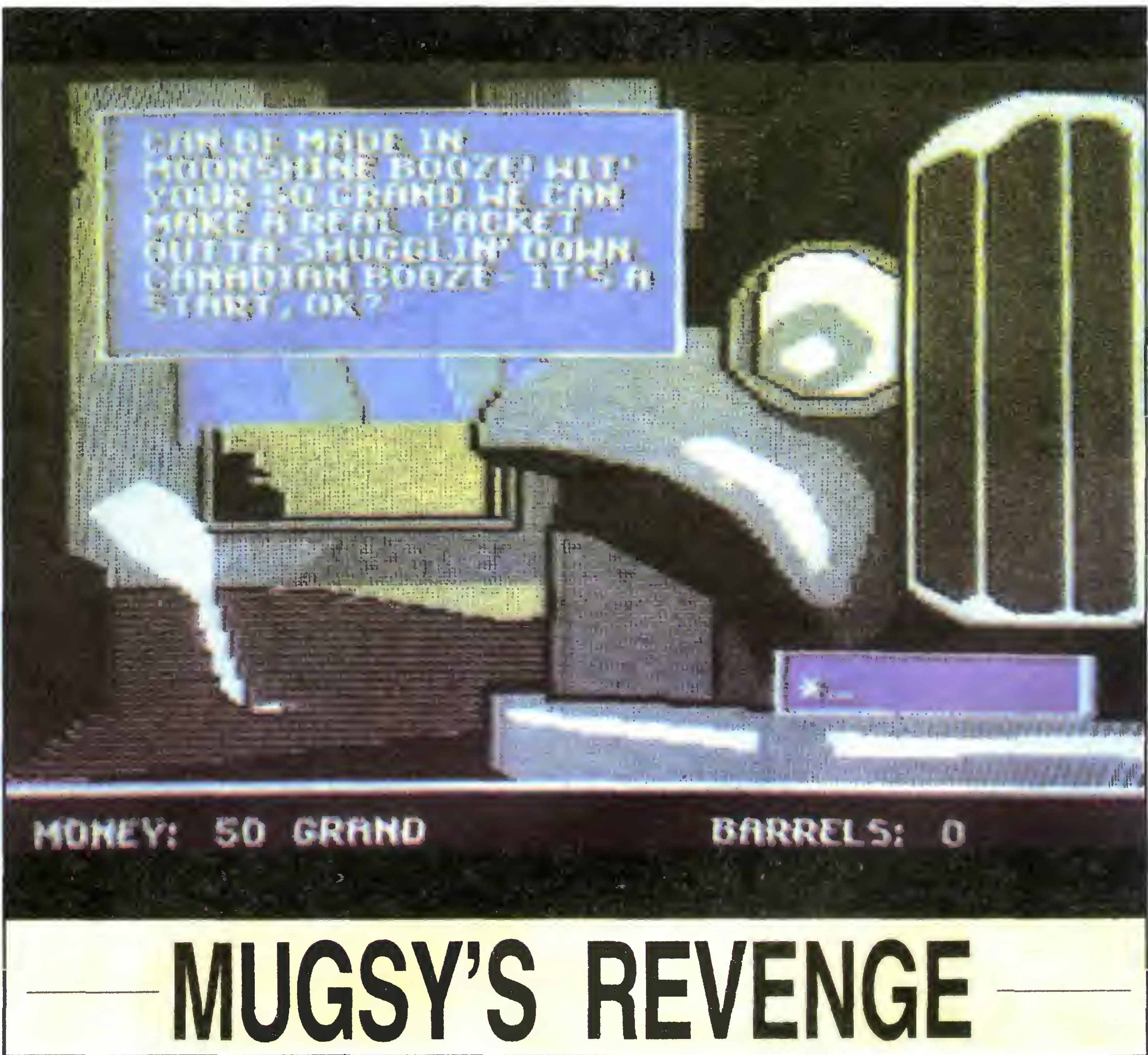
Batman wg nowego filmu (już pewnie nie tak nowego). Red Heat — drugi pomysł z ekranu (Arnie w roli głównej). AFT, czyli komputerowy podręcznik sztuki pilotażu Chucka Yeagera. SkateBall — nowa dyscyplina: łyżwy + kaski + piłka + bramki + przeszkody. Reszta gier znana, a jeśli nie, to poznamy je wkrótce.

Tym razem tylko 1025 propozycji na 94 tytuły.

			ATARI	AMSTRAD	COMMODORE	SPECTRUM
1	BATMAN	↑	>	<	×	×
2	R-TYPE	↓	>	<	×	<
3	TOTAL ECLIPSE	↓		<	<	<
4	ROBOCOP		>	<	×	×
5	GAME OVER II	↑	×	×	<	×
6	AFT	!	>	×	<	<
7	OPERATION WOLF	↓	>	<	×	×
8	RED HEAT	!	>	<	<	×
9	TRAZ	↓	>	<	<	×
10	SKATEBALL	!		<	<	<



Kopertę nadesłał Daniel Pastuszko z Warszawy



MUGSY'S REVENGE

Gra ta przenosi nas na mroczne ulice Nowego Jorku, w początek tego stulecia. Miasto roi się od różnych podejrzanych typów, którzy chcą zarobić szybko i łatwo. Tutaj nie istnieją takie słowa jak uczciwość i wierność. Liczy się spryt, bezwzględność, a przede wszystkim pieniądze.

Uruchamiając tę grę, wcielasz się w postać tajemniczego bilardzisty, o krótkim imieniu Mugsy. Posiadasz tylko 50 GRAND-ów czyli 5000 dolarów oraz listę swoich wspólników. Interes polega na szmuglowaniu beczek z narkotykami tajnymi kanałami do Kanady.

Najpierw musisz zatrudnić tragarzy. Jeden z Twoich wspólników proponuje Ci kilku — najlepiej wybrać jednego, ale pewnego. Często zdarzają się zwykli złodzieje, przez których tracisz mnóstwo pieniędzy. Zwolnienie oszusta zwykle wiąże się z dużym ryzykiem. Po wyrzuceniu grozi Ci on, że powróci z „przyjaciółmi". Często groźba ta zostaje spełniona i na ekranie ukazuje się śmietnik oraz miły napis „GAME OVER".

Po tej trudnej decyzji, następna. Należy zakupić odpowiednią ilość beczek, w których będziesz szmuglował narkotyki. Teoretycznie znajdują się w nich solone ryby, płacisz więc najwyżej 1—2 GRAND-y. Musisz zostawić sobie około 20 GRAND-ów na wypłaty i łapówki.

Gdy posiadasz już baryłki, należy jeszcze „przekonać" do operacji kilka osób. Najpierw policję stanową, której zapłać 10 GRAND-ów. Potem dla posterunków 5 i dla tragarzy sumę, jakiej zażądają. Pozostałe kilkaset dolarów zostaw na czarną godzinę.

W momencie dotarcia na przystań nie kończą się Twoje kłopoty. Czasami napadają Cię rabusie, co kończy się gorącą lufą pistoletu i zimnymi głowami nieboszczyków. Niebezpieczni są również kontrahenci kupujący beczki — często oszukują, rzadziej uczciwie załatwiają interes. Jeżeli masz szczęście i wszystko pójdzie po Twojej myśli — zarobisz na następną transakcję. Jeżeli nie, napis „WE'RE BROKEN" i widok kotów śmietnikowych zakończy grę.

Mugsy, który jest urodzonym abstynentem, pija tylko sok pomarańczowy. Dzięki temu nie obudził się jeszcze z nożem w plecach i kulą w głowie. Czasami jednak ma on sen, przypominający mu młode lata, gdy był zawodowym bilardzistą. Wygrał wtedy dużą sumę od nieznanych ludzi, lecz ci pobili go i ograbili. Od tego czasu nasz sympatyczny bohater przestrzega zasady „ufaj każdemu lecz trzymaj go na muszce". Jeżeli Ty, graczu, będziesz się tego trzymał — zwyciężysz nie tylko na komputerze ale i w rzeczywistości. Powodzenia!

Komputer: ZX Spectrum 48, Commodore 64

Luke

KRÓL I KRÓLOWA GIER



Adam Janczak, lat 12.
VI klasa Szk. Podst. nr 300 w Warszawie
komputer: Toshiba
ulubiona gra: Fly, Szachy
hobby: tenis, narty, język niemiecki i angielski



Małgosia Kalisz, lat 11.
V klasa Szk. Podst. nr 32 we Wrocławiu
komputer: Zx Spectrum
ulubiona gra: Dynamite Dan
hobby: muzyka



AIRBORNE RANGER

Nikt, oprócz niektórych oficerów wojskowych nie wiedział, po co zbudowano tę bazę. Położona z dala od wszelkich miast, otoczona drutami pod napięciem, zaminowana na całym obwodzie, doskonale chroniła się od nieproszonych gości. Wszystkie próby zmierzające do odkrycia sekretu kończyły się z reguły na polach minowych lub w głębokich dołach obozowych.

W rzeczywistości ośrodek ten był miejscem szkolenia komandosów oraz szpiegów politycznych. Uczniowie mieli w przyszłości działać na głębokich zapleczach przeciwnika, niszcząc ściśle określone cele wojskowe. Kandydaci byli szkoleni w zakresie walki wręcz, strzelania, podkładania ładunków itp. Każdy z nich mógł samodzielnie wykonać najtrudniejsze zadanie w bardzo krótkim czasie i cało powrócić do macierzystej bazy.

Wyposażenie takiego sabotażysty składało się zwykle z karabinu maszynowego, pistoletu półautomatycznego oraz granatów ręcznych. Posiadał także ładunki wybuchowe, ustawiane na krótkie czasy, komplet noży i miotacz ognia. Ponieważ na tyły przeciwnika dostawał się zwykle skacząc na spadochronie, więc większość broni i amunicji zrzucano na osobnych spadochronach. Powrót z akcji zawsze następował helikopterem, który o ściśle określonej godzinie czekał w oznaczonym wcześniej miejscu.

Często przeciwnik był informowany o przylocie „gości" i odpowiednio się do tego przygotowywał. W rowach czaili się strzelcy wyborowi, bunkry prowadziły ciągły ogień. Pola minowe i zapory z drutów kolczastych często przegradzały drogę. Gęsto padały rzucane bomby i granaty.

Mimo to nigdy żaden komandos nie wycofał się przed wrogiem, wręcz przeciwnie — nawet atakowali. Ich strzały skutecznie ostudzały przeciwników, a rzucane granaty na długo oczyszczały rejon.

A może Ty chciałbyś wcielić się w postać jednego z takich komandosów? Zapewniam Cię, że nowa gra firmy Micro Prose „Airborne Ranger" dostarcza dużo więcej przyjemności niż podobne do niej na pierwszy rzut oka strzelaniny. Każdy oddany strzał, rzucony granat, podłożona bomba, pozwala pozbyć się nadmiernej ilości zbędnej energii, czasem nawet rozsadzającej Cię. Gdy zobaczysz padającego przeciwnika lub szczątki wysadzonego bunkra, będziesz szczęśliwszy niż po połamaniu domowego fotela.

Przemieniając się w odważnego sabotażystę, możesz wybrać jedną z kilkunastu misji. Każda z nich toczy się na różnych terenach, inni są też przeciwnicy. Masz możliwość niszczenia radarów, domów, porywania oficerów, podkładania bomb w samolotach itp. Zakończenie jednej misji procentuje zawsze awansem na wyższe stanowisko. Wprawdzie nikt nie został jeszcze generałem i podobno dla szczęśliwca, który to osiągnie czeka nagroda. Myślę, że warto spróbować szczęścia.

Firma: Micro Prose
Komputer: ZX Spectrum 48, Commodore 64 IBM PC

Luke

Nowości

Co nowego planują producenci gier? Czym przywitają początek nadchodzącego roku? Oto pigułka gier, które pojawią się na angielskim (a więc wkrótce i na naszym) rynku do 1 stycznia 1990 roku:

— **MYTH** firmy System 3 — doskonała graficzna oprawa, cztery doładowywane levele, wędrówka przez świat żywcem z mitologii greckiej. Odbierająca dech muzyka.
— **VENDETTA** również firmy System 3 — połączenie Commando, Last Ninja, Roadblasters i Cobra Stallone. Niezła grafika 3D, lecz mało kolorowa.
— **GHOSTBUSTERS II** firmy Foursfield — fabuła oparta na filmie pod tym samym tytułem. To już trzecia gra z serii „Ghostbusters", jeszcze bardziej urozmaicona, niż „The Real Ghostbusters".
— **GHOULS'N GHOSTS** firmy US GOLD — nawiązanie do super-hitu „Ghosts'N Goblins" sprzed trzech lat. Znów porwana księżniczka, waleczny rycerz i hordy potworów. Efekty dźwiękowe 128.
— **INTERNATIONAL DRUG BUST** — firmy Players — prawie „Operation Wolf", lecz o mniej urozmaiconej grafice i nieco mniejszej liczbie doładowywanych leveli. Fabuła również inna, ale cel ten sam: ZABIJ!
— **GAŁAXY FORCE** firmy Activision — znana z innych komputerów, tym razem przeniesiona na Spectrum przez Keitha Berkhilla. Oślepiająca grafika standardu Savage i R-Type. Akcja mało skomplikowana, ale dynamiczna i rozluźniająca.

Niestety, zabrakło miejsca na fotografie. Następnym razem na pewno się zmieszczą.

Gen

S.O.S.

Co trzeba zrobić w grze FORCE SEVEN po straceniu amunicji i o co chodzi w grach THANATOS, EIDOLON, RAID 2000 w wersji na Commodore 64?
Tomek Fijakowski, ul. Fornalskiej 4 m 10 26-600 Radom

Od wielu miesięcy szukam gry Tetris na Timexa. Czy wersja taka istnieje?
Artur Baturo, Marcinowice 31 66-600 Krosno Odrz.

Proszę o opisy gier: REVOLUTION, SPITFIRE 40. Jak sterować w LAST NINJA II. W zamian oferuję opisy do: COMMANDO, SABOTEUR, URBAN UPSTART.
Paweł Polesiak, Wyszogrodzka 3 m 34 03-337 Warszawa

Bardzo proszę o opisy do gier: New York City, Nexuss, Mankala. Jak uruchomić grę Alley Cat? W zamian opisy do: Panama Joe, JetSet Willy. Mam Atari 65 XE.
Marta Moraczewska, Jaracza 55a m 33 Łódź

Bardzo proszę o pomoc, poszukuję opisu i mapy do gier PLATOON oraz MIGHTY MAGUS na Timexa.
Michał Bartoszak, ul. Ogrody 11/6 85-870 Bydgoszcz

Poszukuję opisów do gier INFILTRATOR, FEUD, BASIL i MASTERS OF LAMPS na Atari 65 XE. Nie wiem także, jak przejść trzecie jeziorko w grze TIRES.
Ewa Wojtyna, Oświęcimska 85/22 41-707 Ruda Śl. 7

Mam Atari 65 XE. Poszukuję gier PLATOON, HACKER, THE TRAIN, COMMANDO oraz opisu do RAID OVER MOSCOW.
Maciej Dubelewski, Godebskiego 5/16a 81-134 Gdynia

Mam 13 lat i dysponuję komputerem Atari 65 XE. Pilnie poszukuję gier ROAD RACE, BARBARIAN oraz PLATOON. W zamian oferuję opisy do: HACKER, GHOST CHASER, STARQUAKE wraz z grami.
Krystian Dobkowski, Szekspira 2/388 01-913 Warszawa

Szukam gier FIST II, SILENT SERVICE, WINTER GAMES, LAST NINJA, SPY VS SPY, ZORRO. Posiadam Spectrum 48.
Radosław Rusek, Gołębia 27/1 85-309 Bydgoszcz

Posiadam mikrokomputer Amiga 500. Proponuję kontakt w celu wymiany doświadczeń i programów.
Piotr Pacer, Sucharskiego 21b/4 81-157 Gdynia

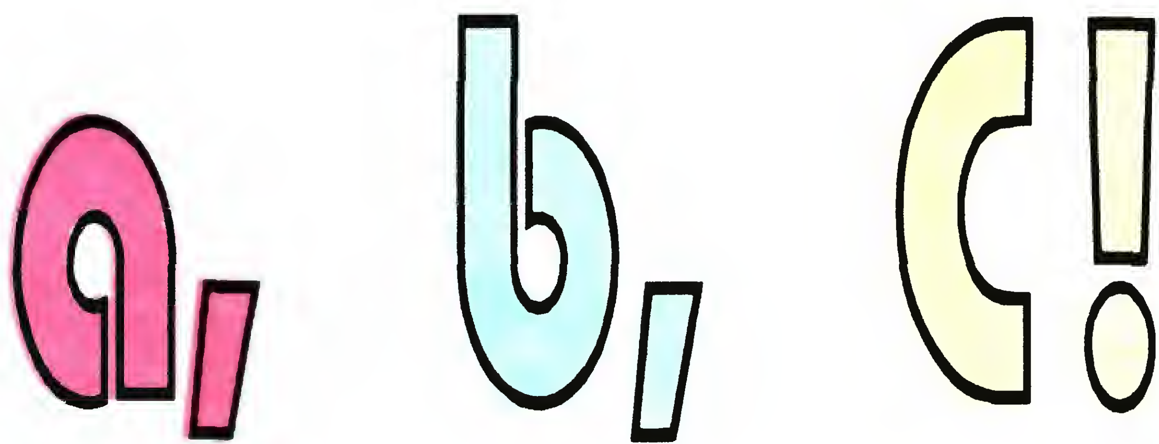
Jestem posiadaczką Atari 65 XE. Pilnie poszukuję następujących gier: Futbol I i II, Bye Maja, Mouse Trap. W zamian ofiaruję inne gry.
Ewa Oleksial, ul. Kniewskiego 2/11 42-500 Będzin

Pilnie szukam asemblera GENS3 i monitora MONS3 na Spectrum. W zamian oferuję kompilatory języków: C i Pascal HP4TM16 lub gry Barbarian, Mikie i Renegade.
Mariusz Bystrzejewski, ul. Westerplatte 12b/15 66-620 Gubin

Mam 10 lat i posiadam Atari 800 XL. Bardzo chciałbym mieć gry Barbarian I i II, The Train, Ace of Aces, World Soccer. Do wymiany przeznaczam kasetę z: Kapitan Beeble, Super Cobra, Biorytm, Submission, Szachy, Bruce Lee, Super Pacman.
Michał Libera, ul. 1000-lecia 35/41 41-308 Dąbrowa Górnicza

Mam Commodore 116 i nie mogę zdobyć interfejsu do tego komputera. Bardzo proszę o pomoc, choćby adres lub schemat.
Jarosław Gierszewski, ul. ZWM 7a/29 75-342 Koszalin

Pilnie poszukuję gier Barbarian, Winter Games, Super Huey, Battle for Normandy, Gauntlet, Kung-Fu Master w wersji kasetowej na Atari 800 XL
Dariusz Bytniewski, ul. Modrzewiowa 24c/16 40-170 Katowice



Każdy, kto choć trochę interesuje się informatyką, bez trudu wymieni kilka lub kilkanaście nazw języków programowania — nie licząc różnych dialektów.

Ot, choćby Action!, Ada, Algol, Basic, C, Comal, Cobol, Forth, Fortran, Lisp, Logo, Modula, Occam, Pascal, PL/I, Prolog, Simula — nie wspominając nic o językach najniższego poziomu, czyli asemblerach. Takie bogactwo różnych sposobów zapisu, lub różnych filozofii stojących u podstaw różnych języków, wiąże się nie z poszukiwaniem języka uniwersalnego, nadającego się do wszystkiego, bo takiego być nie może, ale z mnogością zastosowań. Każdy wymieniony powyżej język ma swoje plusy i minusy, każdy ma swój zakres stosowalności, i — co równie ważne — do pewnych zadań każdy z nich jest niezastąpiony.

Chcę Wam zaproponować poznanie języka C, o którym mówimy dopiero od niedawna. (Dzieckiem tej samej filozofii, co C jest Action!, ale istnieje on wyłącznie na Atari).

Być może jest to związane z tym, że C to język trudny, wymagający znajomości wielu pojęć nieznanych wytrawnym specjalistom od Basica, czy nawet ekspertom od asemblera. Tym niemniej warto zainwestować trochę trudu w nauczeniu się przynajmniej podstaw. Tym, którzy nie wierzą, że warto, dedykuję program, który drukujemy obok. Dlaczego? Otóż jest to nieodłącznie związane z filozofią tego języka.

C wymyślono, żeby ułatwić życie wszystkim tym, którzy wcześniej używali asemblera. Dlatego też zawiera on wiele instrukcji niemal żywcem wyciągniętych z zestawu instrukcji mikroprocesora. Równocześnie jednak jest to język wysokiego poziomu, pozwalający na korzystanie z wyrafinowanych technik programowania — procedury (a właściwie funkcje), skomplikowane struktury danych, wskaźniki — wszystko czego tylko dusza zapragnie. Stąd też obecnie jest to jeden z najpopularniejszych języków stosowanych przez zawodowych programistów. C króluje podczas pisania programów systemowych, edytorów tekstu, gier — do niedawna robiono to wyłącznie przy pomocy asemblera, dzisiaj w asemblerze pisze się najwyżej kilkadziesiąt linii programu — tam gdzie trzeba skorzystać ze szczególnych możliwości mikroprocesora, gdy potrzebna jest maksymalna szybkość lub dokładna znajomość czasu trwania jakiegoś podprogramu w taktach zegara. Kompilatory C istnieją na niemal wszystkich komputerach — od Spectrum do Craya. To też świadczy o jego popularności.

Gra, którą widzicie obok, ma Was przekonać o tym, że to wszystko, co napisałem powyżej, jest prawdą. Jest to gra graficzna, skądinąd bardzo prosta, ale spróbujcie ją napisać w innym języku. Kilka razy skorzystałem przy jej pisaniu z asemblera, ale wyłącznie po to, by móc się odwołać do procedur zawartych w ROM-ie, oraz do rejestru R i do instrukcji IN i OUT procesora. Cała reszta to C. Nawet duszki dały się oswoić (z niewielką pomocą ROM-u). Jeżeli nie znacie C firmy HiSoft, służę kilkoma radami praktycznymi:

1. Po wczytaniu kompilatora naciskamy EDIT i ENTER — przechodzimy do edytora. Edytor jest taki sam jak i w innych kompilatorach HiSoftu (HP4S, GENS).
2. Po wprowadzeniu programu i nagraniu go na taśmę (można również skorzystać z microdrive-u, podając przed nazwą zbioru numer urządzenia poprzedzony dwukropkiem, np. 1:GRA_W_C), piszemy C i ENTER — jesteśmy w kompilatorze. Teraz piszemy #include i ENTER. Po zakończeniu kompilacji naciskamy SYMBOL SHIFT i I, żeby dać kompilatorowi sygnał, że to już wszystko — i gramy.

Sama gra jest prosta, choć nie wróżę nikomu wysokiego wyniku (nasz redakcyjny rekord wynosi 47 punktów). Przy pomocy klawiszy h, n, k oraz o poruszamy po ekranie ośmiokątną ramkę. Ramka służy do zalepiania dziur, które co jakiś czas pojawiają się na ekranie. Żeby dziurę zalepić wystarczy na nią najechać. Ramki wolno używać, dopóki dysponujemy energią. Energia ucieka przez każdą niezalepioną dziurę, ale można ją stracić również wpadając na krawędź ekranu. I ostatnia rada — nie zbliżajcie się zbyt blisko do paskudnej mordy krążącej po ekranie. Jej powolność może okazać się zwodnicza!

Jeżeli chcecie do gry używać Kempstona, wystarczy zmienić definicje lewo, prawo, góra, dół na następujące:

```
#define prawo ( in(31) & 1 )
#define lewo ( in(31) & 2 )
#define dół ( in(31) & 4)
#define góra ( in(31) & 8 )
```

W taki sam sposób można wprowadzić do programu dowolny inny joystick, lub — jeśli komuś na tym zależy — przedefiniować klawisze używane podczas gry.

Napisanie takiej gry jest na Spectrum możliwe tylko przy użyciu C albo asemblera. Program w C powstał w ciągu niecałego tygodnia — wieczorami, po dobranocce, o ile rodzina nie zabrała mi telewizora. Zrobienie tego samego w asemblerze w tak krótkim czasie byłoby niemożliwe — i to jest najważniejszy argument na rzecz C. Naprawdę warto się z nim zapoznać. Zainteresowanych odsyłam do dwóch książek: pierwsza to „Język C" B.W. Kernighana i D.M. Ritchiego — znakomite skrzyżowanie podręcznika z przewodnikiem, nadająca się dla każdego, kto chce poznać **C**; druga — „Sam na sam z językiem C" J. Bieleckiego — zawiera opis (niekompletny) „naszego" kompilatora. Oprócz tych książek radzę przeczytać cykl „Język C dla najmłodszych" — pierwszy odcinek był w Bajtku 6/89.

Marcin Borkowski

```
/*********************************/
/*    Gra w C dla niedowiarków.  */
/*       (c) Bajtek 1989         */
/*  kompilator HiSoft - Spectrum */
/*********************************/

#list-

#define border4 inline(0x3E, 4, 0xCD, 0x229B)
#define cls     inline(0xCD, 0xD6B)
#define halt    inline(0x76)
#define prawo   ( !( in(49150) & 4 ) )
#define lewo    ( !( in(49150) & 16) )
#define dół     ( !( in(32766) & 8 ) )
#define góra    ( !( in(57342) & 2 ) )
typedef char *ChrPtr;

/* W nazwach zmiennych - litera r odnosi się do */
/* ramki, litera m do mordy, litera p oznacza   */
/* zmienną pomocniczą, przechowującą poprzednią */
/* wartość - potrzebną do odtwarzania tła,      */
/* litera k - to kierunek ruchu, v - prędkość.  */

struct sprite {int rozm; ChrPtr s,m; } mp,rp;
char rej_a,dziury[704],ramki,mkx,mky,rndx,rndy;
int energia,ndziur,punkty,mx,my,pmx,pmy;
int rx,ry,prx,pry,rvx,rvy,de,hl;


/* Tablice do zapamiętywania tła.     */
char rmem[48];
char mmem[36];
/* Dane określające wygląd spritów.   */
char rdat[]={7,224,0,15,240,0,28,56,0,56,28,0,112,
14,0,224,7,0,192,3,0,192,3,0,192,3,0,192,3,0,224,7,
0,112,14,0,56,28,0,28,56,0,15,240,0,7,224,0,1,248,
0,3,252,0,7,14,0,14,7,0,28,3,128,56,1,192,48,0,192,
48,0,192,48,0,192,48,0,192,56,1,192,28,3,128,14,7,
0,7,14,0,3,252,0,1,248,0,0,126,0,0,255,0,1,195,128,
3,129,192,7,0,224,14,0,112,12,0,48,12,0,48,12,0,48,
12,0,48,14,0,112,7,0,224,3,129,192,1,195,128,0,255,
0,0,126,0,0,31,128,0,63,192,0,112,224,0,224,112,1,
192,56,3,128,28,3,0,12,3,0,12,3,0,12,3,0,12,3,128,
28,1,192,56,0,224,112,0,112,224,0,63,192,0,31,128};
char mdat[]={7,224,0,63,252,0,115,206,0,97,134,0,225,
135,0,243,207,0,252,63,0,240,15,0,96,6,0,112,14,0,56,
28,0,7,224,0,1,248,0,15,255,0,28,243,128,24,97,128,
56,97,192,60,243,192,63,15,192,60,3,192,24,1,128,28,
3,128,14,7,0,1,248,0,0,126,0,3,255,192,7,60,224,6,24,
96,14,24,112,15,60,240,15,195,240,15,0,240,6,0,96,7,
0,224,3,129,192,0,126,0,0,31,128,0,255,240,1,207,56,
1,134,24,3,134,28,3,207,60,3,240,252,3,192,60,1,128,
24,1,192,56,0,224,112,0,31,128};



main()
{
   char rnd();

   mp.rozm=12; mp.s=&mdat[0]; mp.m=&mmem[0];
   rp.rozm=16; rp.s=&rdat[0]; rp.m=&rmem[0];

   start:

      punkty=0; ramki=5; border4;
      do
      {
         fill(&dziury[0],704,0);
         fill(rp.m,3*rp.rozm,0);
         fill(mp.m,3*mp.rozm,0);
         cls; deska(); opisramek();
         rx=prx=128; ry=pry=85; rvx=rvy=0;
         mx=my=pmx=pmy=80;
         energia=1000; ndziur=0;
         do
         {
            writeat(1,1,16); fprintf(1,"%4d",energia);
            rndx=1+ rnd() % 30;    ruchramki();
            rndy=1+ rnd() % 19;    ruchmordy();
            zalepianie();
            if (rnd() > 124) przedziuraw();
            if ((abs(rx-mx) < 26) && (abs(ry-my) < 16))
                                   atakmordy();
         }
         while (energia>0);
         pauza(50);
      }
      while (ramki--);
      pauza(100);
      writeat(2,11,4);
      printf("Naciśnij dowolny klawisz.");
      while ( !keyhit() );
```

```
   goto start;
}

ruchramki()
{
   char in();

   rvx += prawo-lewo;      rvy += góra-dół;
   rx += rvx;              ry += rvy;
   energia -= ndziur*(punkty>10 ? punkty/10 : 1) -1;
   if (rx>246)  { rx=243;  szum(); }
   if (rx<8)    { rx=11;   szum(); }
   if (ry>167)  { ry=165;  szum(); }
   if (ry<18)   { ry=20;   szum(); }
   putsprite(rx, ry, &rp, &prx, &pry);
}



ruchmordy()
{
   putsprite(mx, my, &mp, &pmx, &pmy);
   if (mkx) mx++; else mx--;
   if (mky) my++; else my--;
   if (mx>243 || mx<13) { mkx=!mkx; beep(100, 100); }
   if (my>168 || my<15) { mky=!mky; beep(100, 100); }
}

atakmordy()
{
   int i;

   energia=0;
   for (i=1;i!=15;i++)
   {
      mx += (mx>rx ? -2 : 2);
      my += (my>ry ? -2 : 2);
      putsprite(mx, my, &mp, &pmx, &pmy);
      beep(5*i, 220-10*i);
   }
}

putsprite(x, y, spr, starex, starey)
   struct sprite *spr; int *starex, *starey;
{
   char i, yc, lyc;
   ChrPtr mp, pmp, sp, psp, pixaddr();

   x -= 8;           y += spr->rozm/2;
   lyc=*starey;      yc=y;
   mp=pmp=spr->m;
   sp= spr->s + 3*((x&6) >> 1)*spr->rozm;

   for (i=0;i!=spr->rozm;i++)
   {
      move(pixaddr(*starex, lyc), mp, 3);
      mp+=3;  lyc--;
   }
   for (i=0;i!=spr->rozm;i++)
   {
      psp=pixaddr(x, yc);   move(pmp, psp, 3);
      *psp++ |= *sp++;      *psp++ |= *sp++;
      *psp |= *sp++;      pmp+=3;      yc--;
   }
   *starex=x;  *starey=y;
}

szum()
{
   int i;

   for (i=0;i!=300;i++) out(254, *cast(ChrPtr)i);
   border4;    energia -= 11;
}


przedziuraw()
{
   int scr;

   scr=32*rndy+rndx;
   if (!dziury[scr])
   {
      ndziur++;                    dziury[scr]=1;
      writeat(2, rndx, rndy);  putc('.', 2);
   }
}

zalepianie()
{
   int scry, scr;   ChrPtr mz;

   scry=-ry/8+21;   scr=32*scry+rx/8;
   if (dziury[scr])
   {
      dziury[scr]=0;
      ndziur--; punkty++;
      mz = &rmem[0] + 1 + 3*(ry - 162 + 8*scry);
      *mz=0;     mz+=3;     *mz=0;
      beep(50, 500);
   }
   writeat(1, 1, 27); fprintf(1, "%4d", punkty);
}

deska()
{
   writeat(1, 0, 0);
   paper(4);    fprintf(1, "%c%c", 6, 6);
   fprintf(1, "          ENERGIA     PUNKTY        ");
   fprintf(1, "%c%c", 6, 6);     paper(7);
}

opisramek()
{
   char i, rmki[7], *rmkip;

   rmkip=&rmki[0];
   for (i=0;i!=ramki;i++) *rmkip++='O';
   *rmkip='\0';
   writeat(1, 1, 2);     fprintf(1, "%-6s", rmki);
}

paper(kolor)
{
   putc(17, 2); putc(kolor, 2);
}

writeat(s, x, y)
{
   putc(22, s); putc(x, s); putc(y, s);
}



pauza(n)
{
   int i;

   for (i=0;i!=200;i+=10) beep(1, 1);
   while (n--) halt;
}

fill(ptr, n, c)
   ChrPtr ptr; int n; char c;
{
   while (n--) *ptr++ = c;
}

int abs(x)
{
   return(x>0 ? x : -x);
}

ChrPtr pixaddr(x, y)
{
   static ChrPtr ptr;

   inline(0xDD, 0x46, 4, 0xDD, 0x4E, 6,
          0xCD, 0x22AA, 0x22, &ptr);
   return(ptr);
}

out(addr, sig)
   char sig;
{
   inline(0xDD, 0x46, 7, 0xDD, 0x4E, 6,
          0xDD, 0x7E, 4, 0xED, 0x79);
}

char in(addr)
{
   inline(0xDD, 0x46, 5, 0xDD, 0x4E, 4,
          0xED, 0x78, 0x32, &rej_a);
   return(rej_a);
}

char rnd()
{
   inline(0xED, 0x5F, 0x32, &rej_a);
   return(rej_a);
}

beep(DE, HL)
{
   de=DE;  hl=HL;
   inline(0xDD, 0xE5, 0xED, 0x5B, &de,
          0x2A, &hl, 0xCD, 0x3B5, 0xDD, 0xE1);
}
#list+
```

REZYDENTNE ROZSZERZENIA SYSTEMU (CP/M PLUS)

W CP/M'ie program użytkownika ładowany do obszaru TPA korzysta z zasobów komputera, oferowanych przez system operacyjny, poprzez odwołania do modułu BDOS. TPA zaczyna się zawsze od adresu 100h, a kończy bezpośrednio poniżej modułu BDOS, który znajduje się w górnych obszarach pamięci.

Przestrzeń adresowa od 00h do FFh wykorzystywana jest przez system na stronę zerową. W 6 i 7 komórce tej strony znajduje się adres modułu BDOS, poprzedzony bajtem C3h (instrukcja skoku bezwarunkowego). W asemblerze odwołania do BDOS'a mają postać CALL 0005h, z uprzednio określonym, w rejestrze C, numerem funkcji.

Standardowo przez BDOS zajęte są numery w zakresach 0-63 i 96—127. Pozostałe wartości mogą być wykorzystane do zdefiniowania własnych funkcji systemowych, wprowadzanych przy pomocy RSX'ów (ang. Resident System eXtension). Dzięki RSX'om możliwa jest także modyfikacja standardowych funkcji BDOS'a.

RSX'y są rezydentnymi rozszerzeniami systemu operacyjnego, których wprowadzenie powoduje zmianę adresu modułu BDOS w komórkach strony zerowej. W takiej sytuacji, po instrukcji CALL 0005, następuje odwołanie do pierwszego, RSX'a, a nie do samego BDOS'a. Dopiero po przechwyceniu odwołania i sprawdzeniu numeru funkcji następuje ewentualny skok do następnego RSX'a, lub do BDOS'a, jeśli był to ostatni RSX.

Rezydentne rozszerzenia systemu umieszczane są bezpośrednio poniżej modułu BDOS i zmniejszają obszar TPA. Ich stosowanie jest koniecznością, jeśli wykonywany kod musi znajdować się we wspólnej części pamięci (adresy C000h-FFFFh). Ma to szczególnie miejsce przy dostępie do środowiska ekranu, które zwykle znajduje się w systemowym banku pamięci.

RSX'y pisane są głównie w asemblerze. Utworzony przy pomocy edytora zbiór musi być poddany asemblacji i konsolidacji. Ostatnią fazę stanowi dołączenie utworzonego RSX'a do programu typu COM.

Konieczne są następujące narzędzia programistyczne:
1. makroassembler relokowalny (najlepiej M80. COM)
2. program konsolidujący (linker) — LINK. COM
3. program GENCOM. COM
4. edytor — do tego celu doskonale nadaje się edytor wbudowany w kompilator Turbo Pascala.

Ogólną strukturę RSX'a przedstawiono na listingu 1.

```
; Nagłówek RSX'a
         DS      6
         JP      Start            ; skok do kodu sprawdzającego
                                  ; numer RSX'a
BDOS:    JP      0000h            ; punkt wejściowy do BDOS'u
         DW      0000h
Remove:  DB      0FFh             ; RSX ma być usunięty
Banked:  DB      00h              ; system ma więcej niż 1 bank
Name:    DB      '01234567'       ; nazwa RSX'a (do ośmiu znaków)
         DB      0,0,0
; początek kodu sprawdzającego numer RSX'a
Start:   LD      A,C              ; pobierz numer funkcji BDOS
         CP      RSXNo            ; sprawdź z numerem RSX'a
         JP      Z,Begin          ; jeśli OK, to skocz do wyk. kodu
         JP      BDOS             ; jeśli nie, kontynuuj dalsze
                                  ; poszukiwania w module BDOS
; początek kodu wykonywanego przez RSX'a
Begin:
;        instrukcje
         RET
; koniec RSX'a
```

Listing 1. Budowa Rsx'a

Najważniejszym elementem jest tutaj nagłówek RSX'a składający się z 27 bajtów. Znajduje się w nim kilka istotnych pól, które zostaną kolejno omówione.

Wewnątrz RSX'a wszelkie odwołania do modułu BDOS, polegają na skoku do etykiety „BDOS", a nie do piątej komórki strony zerowej. Pole opisane etykietą Remove" umożliwia zdefiniowanie RSX'ów, które pozostaną na stałe w pamięci lub zostaną z niej usunięte po wykonaniu programu. Wartość zerowa tego pola definiuje permanentne rozszerzenie systemowe. Pole „Name" służy do zadania 8-znakowej nazwy jednoznacznie identyfikującej RSX'a.

Posłużmy się konkretnym przykładem, aby lepiej zrozumieć budowę RSX'a. Przedstawiony poniżej program w asemblerze umożliwia zarezerwowanie na stałe bufora o pojemności 2KB. Tekst należy wprowadzić do komputera posługując się edytorem. Nazwijmy utworzony zbiór RSV MAC (listing 2).

```
; **********************************
; * RSX rezerwujacy na stale bufor     *
; * o pojemności 2KB. Wywołanie RSX'a *
; * zwraca adres bufora               *
; **********************************
          .Z80
RSXNo     EQU 66              ; numer RSX'a
BufSize   EQU 0800h           ; rozmiar bufora
;
          CSEG                ; segment kodu
; nagłówek RSX'a
          DS     6
          JP     Start
BDOS:     JP     0000h
          DW     0000h
Remove:   DB     0            ; permanentny RSX
Banked:   DB     0
Name:     DB     'Reserve '; nazwa RSX'a
          DB     0,0,0
Start:    LD     A,C
          CP     RSXNo
          JP     Z,Begin
          JP     BDOS
Begin:    LD     HL,Buffer
          RET
Buffer:   DS     BufSize
          END
; **************************************
```

Listing 2. Przykładowy RSX.

Przy założeniu, że na dyskietce znajdują się następujące programy: M80. COM, LINK. COM, GENCOM. COM,
należy wykonać następujące polecenia:
A›M80=RSV
A›LINK RSV [OP]
A›REN RSV.RSX=RSV.PRL
A›GENCOM RSV [NULL]
W wyniku tego postępowania otrzymamy program RSV.COM, którego uruchomienie spowoduje zarezerwowanie bufora. Jego adres będzie dostępny w rejestrze HL, po wywołaniu funkcji modułu BDOS z rejestrem C równym 66. W Turbo Pascalu można to zrealizować przy pomocy jednej instrukcji:
Adres_bufora :=BDOSHL (66);
gdzie zmienna Adres_bufora jest typu INTEGER i została wcześniej zadeklarowana w programie.

Przy pracy z RSX'ami warto zdefiniować następujący zbiór wsadowy o nazwie MAKERSX.SUB (listing 3.).

```
M80 =$1
LINK $1 [OP]
ERA $1.RSX
REN $1.RSX=$1.PRL
GENCOM $1 [NULL]
```

Listing 3. Plik MAKERSX.SUB

Wywołanie: A›SUBMIT MAKERSX.RSV
zastosowane do zbioru RSV.MAC, w obecności programów: SUBMIT.COM, M80.COM, LINK.COM, GENCOM.COM, spowoduje utworzenie zbiorów: RSV.RSX i RSV.COM
Jeśli poprzednio użyliśmy komendy
A›SETDEF [ORDER=(SUB,COM)]
to przygotowanie RSX'a ze zbioru zawierającego jego tekst w asemblerze M80, wymaga następującego polecenia:
A›MAKERSX *nazwa_zbioru_bez_rozszerzenia*

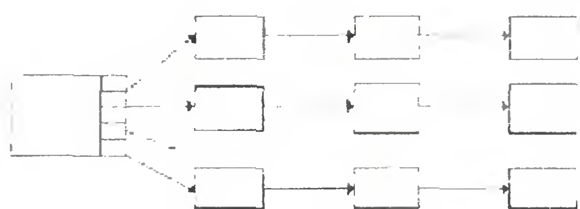
Kolejnym ułatwieniem przy pracy z RSX'ami wykorzystywanymi przez programy pisane w Turbo Pascalu może być dołączenie RSX'a bezpośrednio do kompilatora:
A›GENCOM TURBO.COM *zbiór_typu_RSX*
Oczywiście w przypadku zbioru RSV.MAC i innych permanentnych RSX'ów, nie ma sensu ich dołączanie do programu, ponieważ każdorazowe wywołanie takiego programu instalowałoby kopię RSX'a, prowadząc do stałego zmniejszania obszaru TPA. Permanentne rozszerzenia powinny być tworzone przy pomocy opcji [NULL] polecenia systemowego GENCOM i wywoływane jednokrotnie.
Zbiór typu RSX, otrzymany przy pomocy zbioru wsadowego MAKERSX, może być także dołączony, instrukcją GENCOM, do dowolnego programu:
A›GENCOM *zbiór_typu_COM zbiór_typu RSX*
Jeśli program ten jest napisany w Turbo Pascalu, to kompilując go na dysk musimy pamiętać o zmniejszeniu adresu końcowego pliku wynikowego typu COM. Wynika to z faktu, że dołączony RSX ładuje się jako pierwszy, zmniejszając dostępny obszar TPA, przeznaczony na program użytkownika.

Przykłady RSX'ów w systemie operacyjnym CP/M Plus dla komputerów Amstrad PCW 8256/8512 i CPC 6128 zostały podane w kilku numerach Bajtka w roczniku 1988:
— Nr 2. Informacja o sprzęcie (PCW I CPC)
— Nr 5. Własne znaki na ekranie (PCW)
— Nr 8. Komenda PLOT w Turbo Pascalu (PCW)
— Nr 9. Zrzut ekranu na drukarkę (PCW)
— Nr 10. Własne znaki na ekranie (CPC)
— Nr 11. Rekonfiguracja klawiatury (PCW i CPC)
— Nr 12. Procedury obsługi okien (PCW)

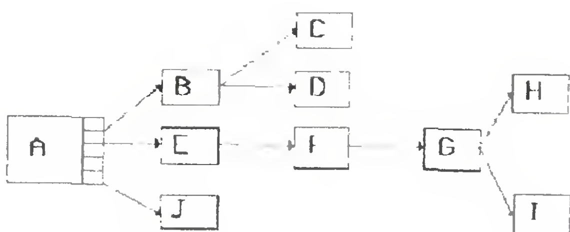
Jonasz Mayer

DRZEWA I GRAFY

Wróćmy na chwilę do naszych rozważań o listach *). Struktura listowa, z samej głębi swej natury, oferuje tylko sekwencyjny dostęp do zapisanych danych. W wielu zastosowaniach jest to duża niedogodność. Wyobraźmy sobie np., że musimy zarejestrować w maszynie nazwiska bardzo licznej, wielotysięcznej, grupy osób. Nawet jeśli uporządkujemy listę nazwisk alfabetycznie, to i tak, za każdym razem gdy zechcemy odszukać kogoś na Z, będziemy musieli przejść przez wszystkie poprzednie litery alfabetu. Dobrze byłoby móc wprowadzić na naszej liście rozgałęzienia, np. oddzielną odnogę na każdą literę
rys.1

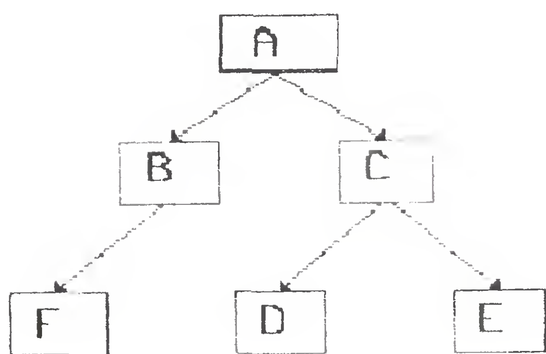


Nie musi to być koniec podziałów. Jeśli nie tylko z pierwszego, ale z każdego elementu listy można tworzyć więcej niż jedną drogę, to otrzymany twór mógłby wyglądać np. tak:
rys.2



Oczywiście nie jest to już lista. Taką konstrukcję zwykle nazywamy drzewem (ang. tree). W drzewie istnieje jeden wyróżniony element, ten z którego (bezpośrednio lub pośrednio) można dojść do wszystkich pozostałych. Ten element to korzeń drzewa (ang. root). Poszczególne elementy drzewa to węzły (ang. node). Do węzła wchodzi dowiązanie z góry, od ojca (ang. parent), i mogą wychodzić powiązania do synów (ang. child). Węzeł, z którego nie wychodzą już żadne dalsze powiązania (który nie ma synów), nazywamy liściem (ang. leave). Kolejne powiązania prowadzące od korzenia poprzez węzły do poszczególnych liści nazywamy ścieżkami lub drogami (ang. path). W drzewie na poprzednim rysunku przykłady ścieżek to: A,B,C; A,B,D; A,E,F,G,I; nie jest natomiast ścieżką ciąg wierzchołków A,B,D,E. Długość ścieżki mierzymy jako liczbę leżących na niej węzłów (łącznie z korzeniem i liściem). Wysokość drzewa to długość najdłuższej ścieżki. Fragment drzewa utworzony w ten sposób, że wybieramy dowolny węzeł i bierzemy wszystkie wychodzące z niego ścieżki nazywamy poddrzewem. Np. w naszym drzewie, jednym z poddrzew jest BDC. Jego wierzchołkiem jest B.

Po tym przeglądzie podstawowych terminów leśniczych jedna drobna uwaga: w większości publikacji informatycznych drzewa rosną korzeniem do góry (czyli zielonym do dołu), a więc tak:
rys.3

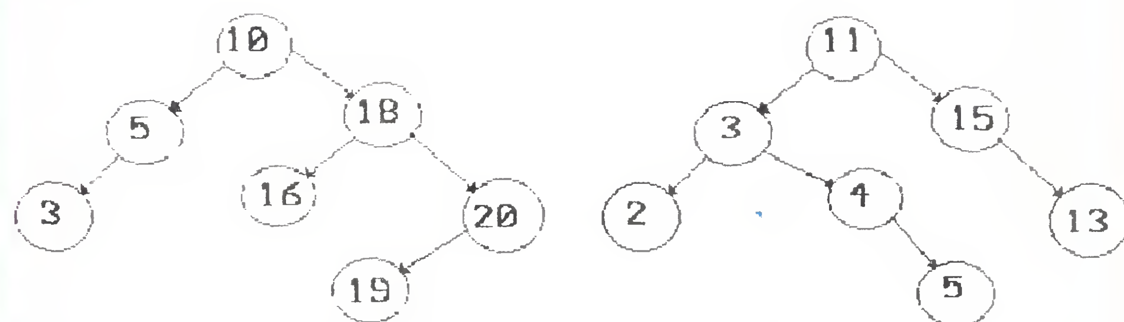


I uwaga poważniejsza. W informatyce używa się przede wszystkim drzew binarnych, tzn. takich, że z jednego węzła wychodzą conajwyżej dwa powiązania. Inaczej mówiąc, w drzewie binarnym wierzchołek może mieć zero, jednego lub dwóch synów.

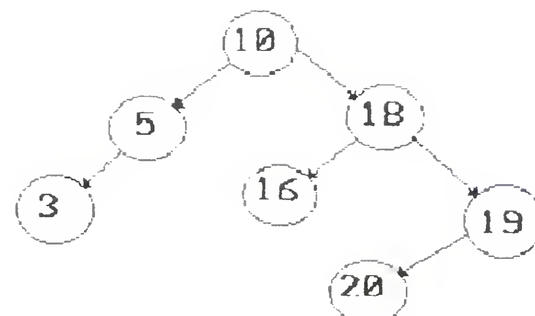
Jeśli chodzi o praktyczną realizację struktur drzewiastych w zwykłej pamięci operacyjnej, to musimy umieć zaimplementować strzałki tworzące powiązania. Można to zrobić w oparciu o zmienne wskaźnikowe (w tych językach programowania, w których zmienne wskaźnikowe istnieją). Można też tworzyć powiązania operując bezpośrednio na adresach obiektów (węzłów) w PaO. Bardziej szczegółowo opisywaliśmy realizację takich powiązań przy tworzeniu list *), nie warto więc chyba tego powtarzać.

Często na strukturę drzewa binarnego nakłada się dodatkowe warunki, a otrzymanej konstrukcji nadaje specyficzną nazwę. Przykładem może być tu drzewo uporządkowane. Jego węzły zawierają liczby, przy czym wykonując wszystkie operacje na tym drzewie musimy zapewnić spełnienie następującego warunku: dla każdego węzła, wartość lewego syna jest mniejsza a wartość prawego syna jest większa niż wartość zapisana w tym węźle.

Oto dwa przykłady drzew uporządkowanych:
rys.4,5



oraz przykład drzewa, które nie jest uporządkowane w opisany wyżej sposób:
rys.6



Spróbujmy w oparciu o takie drzewo rozwiązać raz jeszcze problem, od którego dawno temu rozpoczynaliśmy spotkania ze strukturami danych, czyli zadanie wyszukania podanej wielkości w danym ciągu liczb.

Budujemy drzewo uporządkowane, w którego węzłach znajdują się elementy ciągu. Jeśli teraz chcemy sprawdzić czy liczba X należy do ciągu, to porównujemy ją z wierzchołkiem. Jeśli jest równa to O.K. Jeśli jest mniejsza to przechodzimy do lewego syna, jeśli jest większa przechodzimy do prawego. Powtarzamy to samo postępowanie co w wierzchołku, w efekcie znowu przechodzimy poziom niżej, itd. aż do znalezienia wartości lub dojścia do liścia, skąd nie ma dalszej drogi, co oznacza że X nie występuje w drzewie.

Jeśli budując drzewo oraz wstawiając i usuwając elementy w drzewie już istniejącym spełnimy odpowiednie warunki na równomierność rozbudowy wszystkich gałęzi, to opisany algorytm daje naprawdę rewelacyjne rezultaty. Liczba porównań potrzebnych do znalezienia odpowiedzi jest znacznie mniejsza niż liczba elementów w ciągu.

Na drzewa można patrzeć także nieco inaczej, nie tylko jako na „porozgałęziane listy”. W rzeczywistości ich definicja jest dużo bardziej precyzyjna, gdyż do informatyki drzewa trafiły z matematyki. Stanowią one część większej rodziny obiektów, zwanych grafami. I chociaż nie mam zamiaru wciągać Czytelników w głąb matematyki to chcę również grafom poświęcić chwilę uwagi. Zamiast podawania ścisłej definicji, którą zwykle posługują się matematycy spróbuję odwołać się do intuicji. Zacznijmy od narysowania kilku grafów:
rys.7



Proste, prawda? Graf to nic innego jak zbiór punktów (wierzchołków grafu) połączonych między sobą. Połączenia nazywane są krawędziami grafu, lub (rzadziej) łukami. Krawędzie stykając się w wierzchołkach tworzą drogi. Aby wygodnie było opisywać graf, zwykle wierzchołki oznaczamy, np. literami. Ze względów praktycznych jeszcze lepsze okazuje się często oznaczanie wierzchołków kolejnymi liczbami naturalnymi od 1 do N, czyli po prostu ponumerowanie wierzchołków grafu.

Zwróćmy uwagę, że grafy:
rys.8



to trzy różne obiekty. 1 i 2 mają ten sam zbiór wierzchołków, ale różne zbiory krawędzi. Zaś np. 2 i 3 mają taki sam zbiór krawędzi, ale różnią się zbiorami wierzchołków. Nawiasem mówiąc, 3 jest przykładem tzw. grafu niespójnego, gdyż nie ma w nim możliwości przejścia po krawędziach przez wszystkie wierzchołki. Inny szczególny rodzaj grafu reprezentuje 1 — jest to tzw. graf pełny. Nie można w nim dołożyć żadnej krawędzi, gdyż wszystkie możliwości połączeń między wierzchołkami są już wykorzystane.

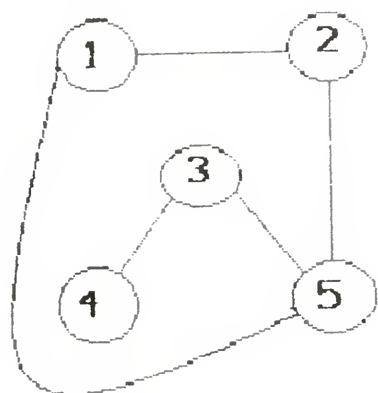
Wspominam tu o grafach mimo tego że nie są one strukturami danych, ale do grafu da się sprowadzić wiele zadań. Inaczej mówiąc, graf jest doskonałym matematycznym modelem dla wielu ważnych problemów praktycznych. Przykładów nie trzeba daleko szukać, wystarczy popatrzeć na mapę połączeń drogowych czy kolejowych, sieci łącznościowe, systemy energetyczne, wodne, kanalizacyjne, czy wiele innych.

Ważnym problemem jest więc właściwa reprezentacja grafów w pamięci, i tym właśnie spróbujemy się zająć. Zakładamy, że graf ma N wierzchołków, ponumerowanych od 1 do N. Jeśli interesuje nas tylko struktura grafu to opis

wierzchołków mamy „z głowy" — nic więcej nie musimy o nich wiedzieć. Gdy z wierzchołkami wiążą się dodatkowe informacje, to możemy je przechowywać np. w tablicy N-elementarnej, przeznaczając po jednym elemencie na każdy z wierzchołków.

Do zapisania połączeń (czyli zbioru krawędzi) grafu możemy użyć tablicy kwadratowej o wymiarach NxN. Nazwijmy ją KRAW [1..N, 1..N]. Dla wartości zmiennych i,j od 1 do N będziemy wpisywać KRAW [i,j]=1 gdy z wierzchołka o numerze j jest krawędź do wierzchołka o numerze i. Jeśli krawędzi między tymi wierzchołkami nie ma, to KRAW [i,j]=0. Oto przykład grafu i zawartości tablicy opisującej jego zbiór krawędzi:

rys.9



	1	2	3	4	5
1	0	1	0	0	1
2	1	0	0	0	1
3	0	0	0	1	1
4	0	0	1	0	0
5	1	1	1	0	0

Metoda jest prosta i naturalna, często z powodzeniem stosowana. Ma jednak również minusy. Tablica o wymiarach NxN ma oczywiście N^2 elementów. Jeśli mamy do czynienia z dużymi grafami, np. mającymi tysiące wierzchołków, to na tablicę KRAW potrzeba milionów komórek pamięci. Nawet samo jej wypełnienie i najprostsze operacje na całym grafie wymagają wykonania ogromnej liczby operacji. Jeśli na dodatek wiemy, że w naszym zastosowaniu występują grafy stosunkowo rzadkie, mające niewiele krawędzi, to reprezentacja przy pomocy tablicy jest zdecydowanie rozrzutna. Przykład: graf ma tysiąc wierzchołków i 5 tysięcy krawędzi. Tablica KRAW zajmuje milion! komórek, z czego tylko 10 tysięcy komórek zawiera jedynki, pozostałe 990 tysięcy musimy pracowicie wypełnić zerami.

W powyższej sytuacji wygodniej jest zapisać zbiór krawędzi grafu w postaci listy. Jeden element listy będzie opisywał krawędź jako parę (p,k) dwóch liczb. Oczywiście liczby te, to numery wierzchołków połączonych krawędzią. Elementów listy będzie dokładnie tyle ile faktycznie istnieje krawędzi.

Pewną odmianą powyższego sposobu jest przechowywanie dla każdego wierzchołka grafu listy krawędzi z niego wychodzących, co właściwie jest równoważne przechowywaniu listy wierzchołków połączonych z danym. Zamiast jednej listy, mamy w tym rozwiązaniu N list — po jednej dla każdego z wierzchołków.

Jak zwykle, określenia która metoda jest najlepsza dokonujemy dopiero znając szczegóły stojącego przed nami zadania.

I wreszcie przyszła pora na dwa pożegnania. Z drzewami i grafami żegnamy się zwracając uwagę, że rozmiary artykułu pozwoliły właściwie tylko zasygnalizować istnienie i wielkie znaczenie tych obiektów. Wiedza na ich temat jest bowiem ogromna. Jeśli ktoś chce uzyskać bliższe informacje musi odwołać się do literatury.

Żegnamy się dziś także z Czytelnikami i sympatykami „Struktur danych", gratulując wytrwałości i dziękując za uwagę.

*) O listach pisaliśmy w „Bajtku" nr i nr

Andrzej Krul

GRAWITACJA

Nigdy jeszcze nie pisaliśmy na łamach Bajtka o zastosowaniu komputerów do symulacji różnych układów, tymczasem jest to jedno z ich podstawowych zastosowań. W wielkich ośrodkach naukowych większość czasu pracy komputerów zajmuje prowadzenie skomplikowanych obliczeń, przy pomocy których można znaleźć rozwiązania zadań nierozwiązywalnych w inny sposób. Chcę Wam przedstawić jedno z tych zadań.

Nazywa się ono problemem n ciał, i jest bardzo proste do sformułowania. Wyobraźcie sobie kilka planet i planetek (czyli n różnych ciał) znajdujących się w znanych miejscach w przestrzeni, poruszających się w znanych kierunkach i ze znanymi prędkościami, oraz działających na siebie siłami grawitacji. Nasze zadanie brzmi: dysponując wszystkimi tymi danymi, obliczyć położenie wszystkich n ciał po upływie określonego czasu (np. za tydzień). Natura daje sobie z tym radę bez problemu, co widać na niebie — tory wszystkich poruszających się po nieboskłonie ciał, takich jak planety, planetoidy, komety i meteory są właśnie efektem rozwiązania problemu n ciał.

To, co robi natura, astronomowie starają się powtórzyć na papierze przy pomocy matematyki. Opisanie przy jej pomocy układu n ciał jest zarazem proste i trudne. Proste — bo prawa, według których te ciała się zachowują, nie są niczym skomplikowanym, zresztą znane są tym z Was, którzy skończyli pierwszą klasę liceum. Trudne — bo matematycy nie potrafią w sposób ogólny rozwiązać powstającego, po wypisaniu wszystkich potrzebnych wzorów układu równań różniczkowych. Mogą sobie z nim poradzić tylko w pewnych szczególnych sytuacjach — kiedy n=2, albo n=3, ale masa jednego z ciał jest tak mała, że można ją pominąć w porównaniu z masami pozostałych ciał — czyli w przypadkach najprostszych. Tymczasem nawet żeby policzyć dokładnie, gdzie będzie się za kilka miesięcy znajdować Księżyc, trzeba wziąć pod uwagę Ziemię, Słońce i przynajmniej trzy planety — Wenus, Marsa i Jowisza, których wpływu nie można pominąć. I co wtedy?

Zanim odpowiemy sobie na to pytanie, obejrzyjmy stosowany przez matematyków i astronomów układ równań. (Zobimy to w uproszczony sposób, tak, byście mogli zrozumieć całe zagadnienie bez wnikania w jego matematyczną stronę, opierającą się na rachunku różniczkowym). Pierwszym równaniem będzie prawo powszechnego ciążenia:

$$\mathbf{F} = -GMm\mathbf{r}/r^3$$

mówiące nam, że siła grawitacji, z jaką dwa ciała się nawzajem przyciągają, zależy od iloczynu ich mas i jest odwrotnie proporcjonalna do kwadratu odległości między nimi. Wprawdzie w mianowniku naszego równania odległość między ciałami jest w trzeciej potędze, ale to dlatego, że siła jest wektorem, to znaczy, że ma nie tylko określoną wartość, ale także kierunek. (Żeby to zaznaczyć oznaczyliśmy ją wytłuszczoną literą). Kierunek siły grawitacji jest dokładnie taki sam jak kierunek wektora łączącego środki obu ciał, a oznaczonego przez **r**. Żeby móc ten kierunek wprowadzić do równania wstawiliśmy **r** do licznika, a r do mianownika. (Kto nie rozumie o co chodzi, niech sięgnie do podręcznika do fizyki dla klasy pierwszej liceum ogólnokształcącego). Jeżeli ciał jest więcej niż dwa, prawą stronę tego równania zastępuje się sumą — wypadkowa siła działająca na ciało jest równa sumie działających sił.

Drugie równanie to drugie prawo dynamiki Newtona:

$$\mathbf{F} = m\mathbf{a}$$

opisujące zachowanie się ciała o masie m pod wpływem przyłożonej siły **F**. Wynika z niego, że jeśli na jakieś ciało działa siła **F**, to będzie się ono poruszać z pewnym przyspieszeniem **a**. Możemy teraz siły występujące w obu wzorach przyrównać do siebie. Otrzymamy następujące równanie:

$$m\mathbf{a} = -\Sigma GMm\mathbf{r}/r^3$$

Masę m możemy skrócić — to co zostanie po prawej stronie równania nazywa się natężeniem pola grawitacyjnego. Wprawdzie w sumie znajdującej się po prawej stronie równania nie ma indeksów, mówiących, które wyrazy trzeba sumować, ale sprawa jest oczywista — dodajemy do siebie natężenia wszystkich pól grawitacyjnych, oprócz pola pochodzącego od ciała o masie m. Ponieważ takie samo równanie możemy napisać dla każdego ciała, mamy układ równań — tyle, ile jest ciał. (Wprawdzie w tych równaniach nie widać nigdzie położenia ciała, ale ono w nich występuje — przyspieszenie jest przecież drugą pochodną położenia po czasie).

Wyobraźmy sobie teraz, co się dzieje w przestrzeni kosmicznej. Weźmy pod uwagę kilka ciał o znanych masach. W pierwszej chwili wszystkie mają jakieś swoje położenie i szybkości, z jakimi się poruszają (są to tak zwane warunki początkowe, od których również zależy rozwiązanie). Ponieważ ciała będą się nawzajem przyciągać, każde z nich będzie poruszać się z pewnym przyspieszeniem, które potrafimy obliczyć na podstawie znajomości mas i położeń. To przyspieszenie spowoduje zmiany szybkości, z którymi ciała się poruszają, i dzięki którym następują zmiany ich położeń. Z kolei zmiana położeń powoduje zmiany odległości między ciałami, a więc i sił z jakimi ciała się przyciągają, co powoduje zmianę przyspieszeń, powodujących zmiany szybkości... — i tak dalej. Gdzie szukać pomocy przy rozwiązywaniu tego galimatiasu?

To już wiemy — w komputerach. Ale jak?

Zwróćmy uwagę na następujący fakt: wprawdzie siła, z jaką ciała się przyciągają, ulega ciągłym zmianom, ale w ciągu odpowiednio krótkiego czasu żadne z ciał nie przesunie się

tak daleko, żeby zmiana siły były duża. Możemy przyjąć, że w ciągu krótkiego czasu (który będziemy oznaczać $\triangle t$) siła jest stała. Skoro tak, to stałe jest również przyspieszenie, i możemy obliczyć nową szybkość ciała po czasie $\triangle t$, jako

Vnowe = Vstare + a△t

(prędkość **v** jest również wielkością wektorową). Skoro się powiedziało a, trzeba powiedzieć również b. Jeżeli w ciągu krótkiego czasu prawie stałe jest przyspieszenie, to szybkość ciała też możemy uznać za stałą, i nowe położenie każdego ciała będziemy obliczać jako

Xnowe = Xstare + v△t

gdzie **x** jest promieniem wodzącym, czyli wektorem łączącym ciało z jakimś wybranym punktem, np. z początkiem układu współrzędnych. W ten sposób możmy obliczyć położenie ciała po czasie △t. A co dalej? Dokładnie to samo — obliczamy nowe wartości przyspieszenia, nowe prędkości i nowe położenia, żeby móc ponownie z nich skorzystać — tak długo, aż otrzymamy upragniony wynik. Oczywiście będzie on obciążony pewnym błędem, ale są różne sposoby na to, by błąd ten był jak najmniejszy.

Teraz czas na napisanie programu, który dokona obliczeń. Aby jeszcze trochę uprościć zadanie, załóżmy że wszystkie ruchy odbywają się w jednej płaszczyźnie (nie musi to być wcale dalekie od prawdy — w układzie słonecznym orbity wszystkich planet leżą z grubsza w jednej płaszczyźnie, tak zwanej płaszczyźnie ekliptyki). We wszystkich wzorach, z których korzystaliśmy, występowały wektory, które teraz rozłożymy na składowe równoległe do osi OX i osi OY. W tablicy **A** znajdują się wartości natężenia pola grawitacyjnego, w tablicy **V** prędkości ciał, a w tablicy **XY** ich współrzędne. Tablica **M** zawiera masy wszystkich **N** ciał.

Wszystkie wartości występujące w programie są przeliczone na nietypowy układ jednostek — masy to wielokrotności masy Ziemi, odległości wyrażone są w jednostkach astronomicznych (1 j.a. to średnia odległość Ziemi od Słońca, czyli około 144 mln km), czas mierzony jest w sekundach. Dzięki temu nie mamy do czynienia z niewygodnymi w użyciu bardzo dużymi liczbami. Wszystkie te zmiany jednostek zostały uwzględnione w stałej grawitacji, która ma w programie wartość 1. 19E—19, różną od znanej nam z układu SI. Prędkość (z km/s) jest przeliczana na j.a./s (służy do tego stała VPRZ). Wartość △t o której podanie program również prosi, musi być wyrażona w sekundach (sensowne wyniki uzyskuje się np. dla △t=100000 s, czyli trochę ponad jedną dobę).

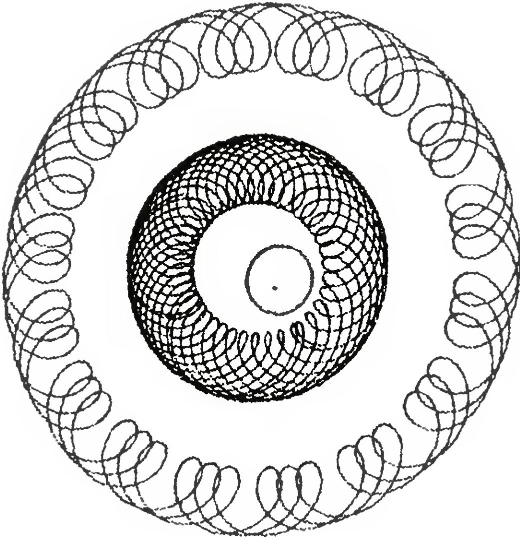
Program napisałem w Pascalu, (żeby przy okazji zachęcić Was do korzystania z tego języka). W procedurze PRZYSPIESZENIA korzystamy w celu skrócenia obliczeń z faktu, że odległość między dowolnymi dwoma ciałami wystarczy obliczyć jeden raz. Stałe występujące w procedurze WYNIKI służą do skalowania rysunku, XM i YM to wyrażone w jednostkach astronomicznych współrzędne lewego dolnego rogu ekranu, DX i DY to ilość pikseli przypadających na jedną jednostkę astronomiczną (aktualne wielkości są dobrane dla Spectrum). Jeżeli podczas obliczeń któreś z obserwowanych ciał znika za krawędzią ekranu, można łatwo zwiększyć obserwowany obszar manipulując opisanymi stałymi. Wprawdzie prezentowana wersja były testowana przy użyciu kompilatora HP4S na Spectrum, ale przeniesienie programu na inne komputery nie powinno sprawić nikomu trudności — przystosowanie programu do kompilatora Turbo Pascala 5.0 zajęło mniej niż pięć minut. Trzy procedury: CLS, KEYPRESSED (zwracająca wartość TRUE jeśli został naciśnięty jakiś klawisz) i PLOT będą wymagały przeróbki, cała reszta programu nie.

Żebyście mogli obejrzeć program w działaniu, proponuję Wam uruchomić go dla następujących danych:

1. Dwa ciała, pierwsze: masa=300000, XO=−1, YO=O, VXO=O, VYO=O; drugie: masa=1, XO=−1, YO=1, VXO=28, VYO=O, delta t = 100000 s. Te dane w przybliżeniu dotyczą Ziemi i Słońca — Słońce stoi nieruchomo, Ziemia krąży dookoła po niemal kołowej orbicie. Zmieńcie teraz masę drugiego ciała (czyli planety) na 10000 — Słońce rusza z miejsca i zaczyna "tańczyć" — na podstawie takiego „tańca" można pośrednio wykrywać planety krążące dookoła innych gwiazd.
2. Trzy ciała, pierwsze: masa 50000, XO=−2, YO=O, VXO=3, VYO=O, drugie: masa 5000, XO=−2, YO=1.5, VXO=10, VYO=−1, trzecie: masa 1, XO=−2, YO=1.6, VXO=21, VYO=−2, delta t 100000. Trzecie ciało będące na początku satelitą drugiego, zostaje przechwycone przez pierwsze, najmasywniejsze ciało (a to wcale nie koniec!). Na tym przykładzie możecie łatwo sprawdzić, że rozwiązanie zależy od przyjętej wartości △t — jeżeli zmniejszycie ją do 10000, historia układu będzie inna. Wiąże się to z zastosowanym przybliżeniem (stałość przyspieszenia i prędkości w ciągu krótkiego czasu). Istnieją sposoby uniknięcia takich kłopotów, ale jest to osobny, bardzo obszerny temat.

Te przykłady to tylko wierzchołek góry lodowej. Ci z Was, którzy interesują się programem na pewno sami znajdą inne ciekawe przypadki. Proponuję im pewną interesującą operację, polegającą na zmianie układu współrzędnych. Jeżeli w instrukcji oznaczonej trzema gwiazdkami oprócz stałych XM i YM umieścicie współrzędne któregoś z obserwowanych ciał, zmienicie układ współrzędnych na związany z tym ciałem (przykładowo zastępując różnicę XY[I,X] −XM wyrażeniem XY[I,X]−XY[1,X]−XM a różnicę XY[I,Y]−YM wyrażeniem XY[I,Y]−XY[1,Y]−YM zmieniamy układ współrzędnych na taki, którego środkiem jest zawsze ciało numer 1). Przy pomocy tej operacji można obejrzeć powstawanie epicykli („pętelek" jakie robią planety na tle gwiazd na skutek ruchu Ziemi). A może program przyda się na lekcjach astronomii?

Marcin Borkowski



Epicykle — kropka w samym środku to planeta, z której prowadzimy obserwacje, kółko dookoła niej to pozorna droga słońca, a „koronki" — to właśnie epicykle.

```
PROGRAM GRAWITACJA;

CONST
  G    = 1.19E-19;
  VPRZ = 6.68E-09;
  MAXN = 50;

TYPE
  KOORDYNATY = (X,Y);
  DANE       = ARRAY[1..MAXN,X..Y]OF REAL;
  MASY       = ARRAY[1..MAXN]OF REAL;

VAR
  A,V,XY : DANE;
  M      : MASY;
  N      : INTEGER;
  DT     : REAL;

PROCEDURE CLS;
BEGIN
  WRITE(CHR(12))
END;

FUNCTION KEYPRESSED : BOOLEAN;
BEGIN
  KEYPRESSED:=INCH<>CHR(0)
END;

PROCEDURE PLOT(X,Y : INTEGER);
BEGIN
  INLINE(#FD,#21,#3A,#5C,#DD,#46,#02,
         #DD,#4E,#04,#CD,#E5,#22)
END;

PROCEDURE PRZYSPIESZENIA(VAR A,XY : DANE;
                         VAR M : MASY;N : INTEGER);
VAR
  DELTX,DELTY,R,RM : REAL;
  K   : KOORDYNATY;
  I,J : INTEGER;
BEGIN
  FOR K:=X TO Y DO
    FOR I:=1 TO N DO A[I,K]:=0;
  FOR I:=2 TO N DO
    FOR J:=1 TO I-1 DO
    BEGIN
      DELTX:=XY[I,X]-XY[J,X];
      DELTY:=XY[I,Y]-XY[J,Y];
      R:=SQR(DELTX)+SQR(DELTY);
      R:=R*SQRT(R);
      RM:=M[J]/R;
      A[I,X]:=A[I,X]-RM*DELTX;
      A[I,Y]:=A[I,Y]-RM*DELTY;
      RM:=M[I]/R;
      A[J,X]:=A[J,X]+RM*DELTX;
      A[J,Y]:=A[J,Y]+RM*DELTY
    END
END;
PROCEDURE PODELTAT(VAR DOZM,WSP : DANE;
                   N : INTEGER;DT : REAL);
VAR
  K : KOORDYNATY;
  I : INTEGER;
BEGIN
  FOR K:=X TO Y DO
    FOR I:=1 TO N DO
      DOZM[I,K]:=DOZM[I,K]+WSP[I,K]*DT
END;

PROCEDURE WYNIKI(VAR XY : DANE;N : INTEGER);
CONST
  YM = -2.0;
  DY =  44.0;
  XM = -2.9;
  DX =  44.0;
VAR
  I : INTEGER;
BEGIN
  FOR I:=1 TO N DO
    PLOT(ROUND((XY[I,X]-XM)*DX),
         ROUND((XY[I,Y]-YM)*DY))                {***}
END;

PROCEDURE WCZYTDANE(VAR N : INTEGER;VAR M : MASY;
                    VAR XY,V : DANE;VAR DT : REAL);
VAR
  K : KOORDYNATY;
  I : INTEGER;
BEGIN
  WRITELN('Masy - wielokrotność masy Ziemi.');
  WRITELN('Odległości - w jedn. astronomicznych.');
  WRITELN('Prędkości - Km/s.');
  WRITE('Podaj ilość obiektów ');
  READ(N);
  WRITELN('Wprowadzaj kolejno dane:');
  FOR I:=1 TO N DO
  BEGIN
    WRITE(I:2,' masa ');  READ(M[I]);
    M[I]:=G*M[I];
    WRITE('XO ':8);       READ(XY[I,X]);
    WRITE('YO ':8);       READ(XY[I,Y]);
    WRITE('VXO ':8);      READ(V[I,X]);
    WRITE('VYO ':8);      READ(V[I,Y]);
    V[I,X]:=VPRZ*V[I,X];
    V[I,Y]:=VPRZ*V[I,Y]
  END;
  WRITE('Podaj delta t ');
  READ(DT)
END;

BEGIN
  WCZYTDANE(N,M,XY,V,DT);
  CLS;
  REPEAT
  BEGIN
    PRZYSPIESZENIA(A,XY,M,N);
    PODELTAT(V,A,N,DT);
    PODELTAT(XY,V,N,DT);
    WYNIKI(XY,N)
  END
  UNTIL KEYPRESSED
END.
```



Dawno, dawno temu, gdy wszyscy zachwycaliśmy się możliwościami wszechobecnego Spectrum 48, nikt nie zwrócił uwagi na model 128 i następne. Bo też nie było w nim prawie nic godnego uwagi.

Prawie nic, czyli wszystko oprócz AY-greka. Z początku tkwił we wnętrzu Spectrum 128 niezauważony. Potem niektórzy przypadkiem usłyszeli jego muzykę. Powoli rodziła się ciekawość.

Wreszcie wszystkich opanował szał. Kilku elektroników naraz opracowało schematy podłączenia AY-greka do starego Spectrum 48 i +. Odkryto, że wiele gier posiada programy obsługi generatora. Zaczęto pisać programy demonstracyjne, z muzyką „wyciąganą" z gier. Dotychczas słowa „Spectrum" i „muzyka" były przeciwnościami, teraz stały się jednością.

Przedstawiamy tu program „The Music Box 128" firmy Melbourne House. Został on napisany na przełomie lat 1985/86 i jak dotąd jest najlepszym (!) z programów do pisania muzyki w Polsce. Podobno z początkiem 1990 roku dotrzeć ma do nas „Music Artist" Benna Daglisha, lecz jest to pod znakiem zapytania.

Music Box 128 napisany został na Spectrum 128 i dlatego przystosowano go do współpracy z magnetofonem, mikrodrajwami oraz RAM-dyskiem (komendy **SAVE!** i **LOAD!**). W prosty sposób przerobić można jedną z tych opcji na współpracę ze stacją **FDD 3000** (system operacyjny TOS), gdyż muzyka zajmuje pamięć niedynamicznie i przydzielone jest na nią ok. 4700 bajtów.

Program składa się z trzech segmentów — Basicowego loadera, bloku kodu i programu zarządzającego w Basicu. Wszystkie operacje sięgają korzeniami w Basic i posiadają odwołania do kodu. W nim wykonywane są jedynie przesłania bloków pamięci, zapis nut i efektów specjalnych oraz odtwarzanie muzyki.

Po wczytaniu programu zgłasza się główne menu, poprzedzone czasem (w niemodyfikowanych przez naszych pseudo-hackerów sprowadzających nowości) muzyczką demonstracyjną.

Menu zawiera pozycje:

1. **LOAD TUNE**
2. **SAVE TUNE**
3. **SYSTEM MENU**
4. **ENVELOPES**
5. **SET TEMPO**
6. **EDIT MODE**
7. **HELP**

Pierwsze dwie opcje służą do zapisu i odczytu kodu melodii z pamięci zewnętrznej. Standardowo ustawiony jest RAM-dysk, więc po uruchomieniu programu na Spectrum 48 i wybraniu jednej z tych opcji, program zatrzyma się komunikatem „Nonsence in Basic".

Trzecia opcja — **SYSTEM MENU** — powoduje zgłoszenie kolejnego menu:

1. **SET PRESET NOISE VALUES**
2. **SET CHANNEL LOOP PARAMETERS**
3. **CHANGE SAVE/LOAD PERIPHERAL**
4. **LISTEN TO TUNE**
5. **COMPILE AND SAVE**

Opcja pierwsza pozwala na zmianę tzw. efektów perkusyjnych, określonych tu jako „noise". Wyświetlana jest lista aktualnie istniejących obwiedni dźwięku oraz charakterystyka poszczególnych uderzeń. Klawisz **0** powoduje powrót do głównego menu. Klawisze **1—9** pozwalają na zmianę odpowiednich uderzeń. Należy podać częstotliwość i rodzaj obwiedni dźwięku.

Opcja druga pozwala na ustalenie miejsc zapętlenia trzech kanałów oraz kanału efektów specjalnych. Wyboru dokonuje się wciśnięciem klawisza **1—4** i podaniu numeru nuty, po której ma nastąpić zapętlenie. Klawisz **0** powoduje powrót do głównego menu.

Opcja trzecia to ustalenie rodzaju pamięci, w której odkładana będzie przez **SAVE** i odczytywana przez **LOAD** melodia. Standardowo jest to RAM-dysk, możliwe jest wybranie mikrodrajwów lub taśmy.

Opcja czwarta odtwarza zawartą w pamięci muzykę, uwzględniając ustalone tempo. Jest to odtwarzanie „rzeczywiste", gdyż w trakcie nie odbywa się obróbka obrazu.

Opcja piąta pozwala na skompilowanie napisanej melodii i nagranie jej w postaci bloku kodu maszynowego, pracującego samodzielnie. Kompilacja nie jest dokonywana, jeśli któryś z kanałów nie był zapętlony.

Wróćmy do głównego menu (zwykle powoduje to klawisz **0**). Opcja **ENVELOPES** umożliwia zaprojektowanie kształtu ośmiu obwiedni dźwięku, które wykorzystywane będą w edytorze melodii. Zmiany kształtu po wciśnięciu numeru obwiedni dokonuje się klawiszami kursora (**5, 6, 7, 8** z **CAPS SHIFTEM**), ustalenie — klawiszem **ENTER**.

Kolejna opcja — **SET TEMPO** — pozwala na ustalenie szybkości odtwarzania melodii. Aktualna szybkość symbolizowana jest poziomym paskiem u dołu ekranu, zmiana długości dokonywana jest klawiszami **5** i **8**. Wciśnięcie innego klawisza powoduje przejście do edytora.

EDIT MODE to właśnie edytor. Nie jest on „user-friendly", ale nie wymagajmy zbyt wiele od programu sprzed pięciu lat!

Najbardziej zrozumiały będzie suchy opis klawiatury:

1, 2, 3, 4 — oktawy
6 — powrót do menu
7 — skasowanie zapisu melodii
Q — odtworzenie melodii
W — ustalenie zapętlenia w aktualnym kanale
R — powrót na początek melodii
T — zmiana kanału
O, P — odtworzenie kolejnego taktu melodii
A, S, F, G, H, K, L — nuty z krzyżykiem, klawisze jak na klawiaturze pianina
CAPS SHIFT, CAPS LOCK, Z, X, C, V, B, N, M — nuty zwykłe, klawisze jak na pianinie
SYMBOL SHIFT + **Q-O** — uderzenia perkusji

Pewnym utrudnieniem dla posiadaczy Spectrum 48 jest układ pianinopodobnej klawiatury. Wykorzystując bowiem układ ze Spectrum 128 zaangażowano cały dolny rząd liter wraz z **CAPS SHIFTEM**, kropką i **CAPS LOCKIEM**.

Na ekranie edytora widać okno z dwiema pięcioliniami oraz okno informacyjne. Podawane są w nim kształty obwiedni w poszczególnych kanałach, głośność również w poszczególnych kanałach, numer odgrywanej nuty z każdego kanału, numer oktawy i numer bieżącego kanału, informacja o zapisie nie będącym nutą ani uderzeniem (**EFFECTS CH**) oraz mały wskaźnik — głośnościomierz.

Wciśnięcie jednego z klawiszy pseudoklawiatury pianina powoduje przesunięcie w lewo pięciolinii oraz ustawienie nowej nuty przy prawej krawędzi ekranu. Cofnięcie pięciolinii umożliwia klawisz **DELETE** (**CAPS SHIFT** i **0**), nie jest on powtarzalny. Wciśnięcie **ENTER** powoduje pozostawienie „pustego" miejsca lub skasowanie istniejącej następnej nuty. Wszystko to odnosi się do aktualnego kanału, którego numer podawany jest w okienku **STATE CHN:**.

Wciśnięcie **EXTENDED MODE** (**CAPS SHIFT** i **SYMBOL SHIFT**) powoduje możliwość wstawienia w zapis melodii specjalnych znaczników. Przedtem wyświetlane jest mini-menu w oknie systemowym, które udostępnia klawisze:

ENTER lub **R** — powrót do edytora
E — ustalenie nowej obwiedni dla jednego lub wszystkich kanałów
V — ustalenie głośności w jednym lub wszystkich kanałach
B — wstawienie „pustej" nuty
S — wstawienie tzw. międzydźwięków („semitones"), powodujących wibrowanie, unoszenie lub opadanie „zwykłych" nut. Podawana jest liczba „drgnięć" i kierunek — góra lub dół
Q — przeniesienie podanego fragmentu zapisu w inne miejsce z zachowaniem fragmentu źródłowego

Na koniec należy podkreślić najpoważniejszą — moim zdaniem — wadę Music Boxa, tzn. stałą długość nut. Wszystkie są ósemkami!

Można starać się symulować nuty dłuższe przez ustawienie obok siebie jednakowych nut o stałej obwiedni, lub dostawianie nuty o obciętej obwiedni do nuty „całej". Można wreszcie pokusić się o modyfikację programu.

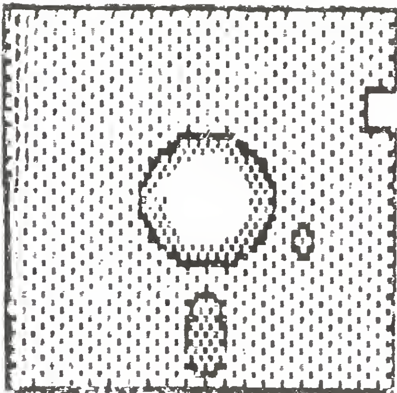
Mimo, że niezwykle trudno jest sklecić coś mile brzmiącego, powodzenie Music Boxa nie słabnie. Z licznych programów muzycznych znany jest Spectrumowcom p. Maciej Wroński, który przy pomocy Music Boxa pisze profesjonalnie brzmiące utwory. Niezorientowanym polecam programy demonstracyjne „DOBRANOCKI" i „KOLĘDY".

Marcin Przasnyski



Ceny mogą być wyższe ze względu na inflację, a poza tym obowiązuje tzw. „przelicznik dolarowy"

	Giełda Bajtka	Sklep Bajtka	Komis	Pewex	RFN	Baltona	CSH i inne
	tys. zł	tys. zł	tys. zł	$	DM	$	tys. zł
SINCLAIR							
ZX 81	120	120	—	—	—	—	—
ZX Spectrum 48	400	350	550	—	80	—	—
ZX Spectrum +	450	550	700	—	90	—	—
Timex 2048	400	400	400	—	—	—	240
ZX Spectrum 128+	550	—	—	—	—	—	—
ZX Spectrum 128+2	—	—	—	—	140	—	—
ZX Spectrum 128+3	—	—	—	—	280	—	—
drukarka Seikosha GP 50s	—	280	—	—	—	—	—
Interface Kempston	25	20	—	—	10	—	9.3
COMMODORE							
Commodore 64	900	650	710	199	290	155	950
VC 20	200	180	—	—	—	—	—
C 16	300	250	—	—	80	—	—
C 116	350	240	—	—	70	—	—
C Plus 4	450	450	380	—	150	—	—
C 128	1100	1100	—	—	399	—	—
C 128 D	1900	—	—	450	820	—	—
Amiga 500	2300	2900	—	—	899	—	—
Magnetofon 1531	150	170	170	48	30	225	—
Stacja dysków Oceanic	700	700	—	—	320	170	—
Stacja dysków 1571	1000	1000	—	199	460	—	—
Drukarka LCIOC	1300	1300	—	—	260	230	—
ATARI							
Atari 800 XL	600	700	470	—	160	—	—
Atari 65 XE	700	650	520	127	—	—	—
Atari 130 XE	800	850	—	199	220	—	—
Atari 520 ST	2000	2000	—	—	—	—	—
Atari 1040 ST	—	—	—	—	1140	—	—
Magnetofon XC 12	140	140	—	36	40	—	—
Stacja dysków 1050	900	900	750	185	300	—	—
Stacja dysków 520 STM	—	700	—	—	—	—	—
Drukarka 1029	400	650	—	—	—	—	—
AMSTRAD							
Amstrad 464 mono.	900	900	—	—	350	—	—
Amstrad 664 mono.	—	1200	—	—	—	—	—
Amstrad 6128 mono.	—	—	—	—	670	—	—
Amstrad PCW 8256	—	—	—	—	—	—	—
Amstrad PCW 8512	—	—	—	—	—	—	—
Amstrad PCW 9512	—	—	—	—	—	—	—
Stacja dysków do 464	—	—	—	—	380	—	—
SHARP							
Sharp MZ 700	—	360	—	—	—	—	—
Sharp MZ 800	—	420	—	—	—	—	—
Dyskietki 5.25 cala	2	2-2.5	3	1	0.7	11	4-9
Dyskietki 3.5 cala	7.5	7.5-9	7-9	—	5	2.5	9-10
Dyskietki 3 cale	11	—	—	—	6	3	10
Joystick	21	16-26	20-30	5	10	10	14
Monitor Neptun	60	80	—	—	—	—	68



INDYWIDUALNY BANK DANYCH

Krzysztof Zając, lat 17. Posiada Commodore C 64 ze stacją dysków 1541 II. Dysponuje zestawem 1600 programów, w tym programy muzyczne, gry oraz graficzne. Proponuje wymianę najnowszych programów. Adres: 43—316 Bielsko-Biała, ul. Roztoki 7/24.

Grzegorz Zakolski, posiada Atari 800 XL, stację dysków LDW 2000, duży zbiór instrukcji i około 900 programów. Zainteresowany jest wykorzystaniem różnych urządzeń peryferyjnych do współpracy z komputerem. Pragnie nawiązać korespondencję w celu wymiany doświadczeń, literatury oraz programów. Adres: 06-100 Pułtusk, ul. Armii Czerwonej 5.

mgr Juliusz Kobylanka — nauczyciel, posiada mikrokomputer Meritum I. Nawiąże kontakt w celu wymiany programów. Adres: 44-244 Żory, oś Sikorskiego, Szkoła Podstawowa 14.

Janusz German prosi o kontakt posiadaczy Atari XL/XE z przerobionym magnetofonem (TURBO 2000, AST) w celu wymiany oprogramowania. Adres: 83-130 Pelplin, ul. Mickiewicza 7.

Maciej Węgorkiewicz, lat 14. Posiada Amstrada CPC464. Pragnie nawiązać kontakt z użytkownikami tego komputera w celu wymiany oprogramowania. Adres: 27-400 Ostrowiec Św., oś. Słoneczne 46/16.

Marek Laszczka, lat 14. Jest świeżo upieczonym posiadaczem Atari 520STFM z monitorem monochromatycznym SM 125. Prosi o kontakt w celu wymiany doświadczeń oraz oprogramowania. Adres: 01-957 Warszawa, ul. Szegedyńska 5/14.

Szymon Wiaderski jest posiadaczem Timexa 2048, oraz około 100 gier i programów użytkowych. Proponuje wymianę programów, literatury, opisów gier. Adres: 82-300 Elbląg, ul. 22 Lipca 13/I/6.

Agnieszka Bukiewicz, lat 17. Posiada mikrokomputer Meritum, chętnie nawiąże kontakt z posiadaczami tego typu komputera w celu wymiany programów i literatury. Adres: 59-800 Lubań Śl., ul. Wojska Polskiego 44 „b"/30.

Tomasz Roszczyk, posiada Atari 520 STFM z podwójną stacją dysków. Nawiąże kontakt z posiadaczami ST. Adres: 41-706 Ruda Śląska, ul. Poniatowskiego 19.

Wladimir Timofeew, lat 25. Posiada mikrokomputer Atari 65 XE z magnetofonem. Oprogramowanie około 100 gier. Nawiąże korespondencję z posiadaczami Atari w celu wymiany gier. Korespondencja w języku polskim lub angielskim. Adres: 195009 Leningrad, ul. Aszenalnaja 70 k 3, ZSRR.

Zofia Filuś, uczennica lat 14. Posiada mikrokomputer Thomson TO 07 oraz cartrige z wersją Basic 1,0. Prosi posiadaczy gier do tego komputera o kontakt. Adres: 41-902 Bytom, ul. Grottgera 22 b/4.

Tomasz Kruklis, posiada Atari 800 XE magnetofon XC 12 programujący w „AST" (Atari Super Turbo). Chętnie nawiąże kontakt z użytkownikami tego systemu w celu wymiany oprogramowania oraz doświadczeń. Adres: 11-400 Kętrzyn, ul. Piastowska 7/37.

Artur Kuliński, lat 17. Posiada Atari 65 XE, stację dysków LDW 2000 oraz magnetofon XC 12. Proponuje wymianę oprogramowania, doświadczeń i literatury. Adres: 37-600 Lubaczów, ul. Kościuszki 85.

Aleksander Walentynowicz Romanow, fizyk, lat 31. Posiada ZX Spectrum 128+, kilka własnoręcznych interfejsów oraz stacje dyków w systemie Beta 128 (TR-DOS ver. 5.01). Prosi posiadaczy podobnych systemów operacyjnych o kontakt w celu wymiany doświadczeń i oprogramowania. Korespondencja w języku polskim, angielskim, ukraińskim lub rosyjskim. Adres: CCCP, 252033, Kijew — 33, ul. Władimirskaja 83 m 6.

Na listy Czytelników odpowiada Dominik Falkowski

Drogi Bajtku!

Mam komputer ZX Spectrum 16 KB. Niestety, tyle pamięci to za mało, żeby wgrać jakiś porządny program, grę. Słyszałem, że istnieją przystawki rozszerzające pamięć. Jak je zdobyć?

Seweryn Ślesicki
Dębno Lubuskie

Spectrum „szesnastka” rzeczywiście jest bardzo ograniczony w możliwościach ze względu na małą pamięć. Z reguły jednak na płytce znajduje się miejsce na dołożenie kości pamięci zwiększających ją do 48 KB. Zabieg ten należy polecić doświadczonemu elektronikowi, obeznanemu z wnętrzem Spectrum. Kości pamięci dostępne są na giełdach i czasem w sklepach elektronicznych.

Chciałbym kupić Spectrum, nie mogę się jednak zdecydować na wybór modelu. 48 raczej odpada ze względu na klawiaturę. Plus jest lepszy, ale ma tylko 48 KB pamięci. Model 128 jest wart zainteresowania, ale istnieją też i modele +2 i +3. Bardzo proszę o pomoc.

Tadeusz Kipisz
Grójec

Podstawową rzeczą przy wyborze komputera, to zadać sobie pytanie, do czego chce się go używać. Jeśli zdecydowany na kupno Spectrum, to będzie Ci on służył prawdopobnie do gier, muzyki i kilku drobnych rzeczy, do których może służyć każdy komputer.

Z tego, co piszesz wnioskuję, że chciałbyś sporą pamięć, dobrą klawiaturę, a nawet stacje dysków. Optymalnym rozwiązaniem wydaje mi się kupno Spectrum Plus (ok. 80 $), zamontowanie w nim AY-greka i interfejsu Kempston do joysticka (ok.15 $), rozszerzenie pamięci do 80 KB (cena zależna od przypadku, czasem wystarczy jeden opornik, czasem trzeba dołożyć kości), kupno stacji dysków FDD 3000. Cena nie powinna przekroczyć 200$, a zatem taki jest pod każdym względem zadowalający.

Mam pewne pytanie, a mianowicie do czego służy program IBM 2, który kupiłem na giełdzie?

Krzysztof Węglarz
Warszawa

Program IBM 2 jest to napisany przez naszego redakcyjnego kolegę, Macieja Pietrasia symulator IBM-a. Zabawa tym programem pozwala nieco przybliżyć architekturę systemu operacyjnego DOS, nakładek GEM i Norton Commander, prace z twardym dyskiem. Tak naprawdę, to IBM 2 służy do niczego, a tylko naśladuje widoczne na ekranie efekty komputera IBM. Można jednak nieźle bawić się przebijając się przez kolejne pomysły autora, a liczba ich zawarta w jednym programie jest ogromna.

Jak podłączyć TV do ZX Spectrum lub Commodore 16? Jakie zastosować kable i złącza?

Darek Slubowski
Szczecin

Zarówno Spectrum jak i Commodore 16 wyposażone są w modulatory telewizyjne. Znaczy to, że mogą generować sygnał wizyjny do odtworzenia na ekranie telewizora. Do połączenia wystarczy koncentryczny kabel antenowy z takimi końcówkami. Telewizor powinien posiadać zakres UKF (21—60).

Jestem posiadaczem komputera ZX Spectrum i mam kilka pytań:
1. W jaki sposób przesyłane są bajty z magnetofonu do komputera?
2. W jaki sposób ATARI SUPER TURBO przyspiesza wczytywanie?
3. Czy możliwe jest zrealizowanie TURBO na Spectrum?

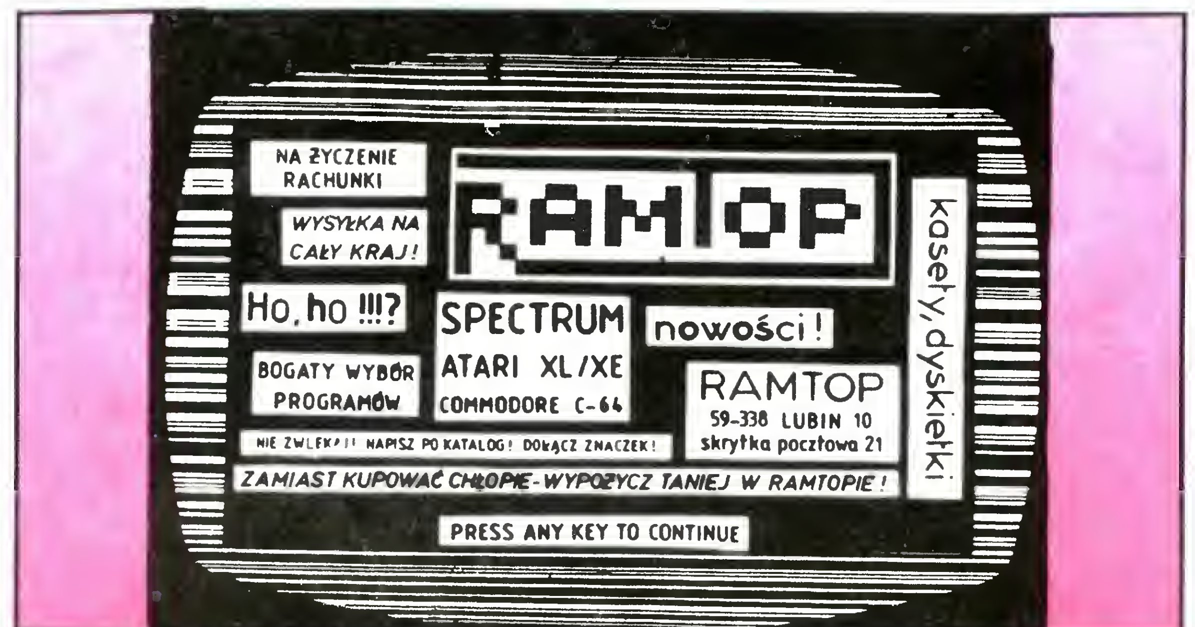
Marcin Chmiela
Kraków

1. Transmisja danych z magnetofonu do komputera odbywa się poprzez dwużyłowy kabel. Informacja zapisana jest na taśmie w postaci pisków. Analiza takich sygnałów ukaże dwa poziomy napięcia w ściśle określonej konfiguracji czasowej. System komputera potrafi identyfikować zmiany napięcia podawane na wejście jako jedności informacji układające się w bajty. Myślę, że o to chodziło Ci w tym pytaniu.

2. Sposób, w jaki ATARI SUPER TURBO przyspiesza wczytywanie jest tajemnicą firmy. Z grubsza polega to na zmianie sposobu zapisu informacji na taśmie. Oryginalny system to bloki po 128 bajtów, każdy z sumą kontrolną. Prędkość ok. 600 bodów to iście żółwie tempo. Po modyfikacji, nowy system zapisu jest mniej rozrzutny i tę samą informację zapisuje na mniejszym odcinku taśmy.

3. TURBO na Spectrum istnieje w kilku wersjach. Jedną z nich jest nakładka na program COPY-COPY, lecz służąca tylko do archiwizacji programów na taśmie. Druga, to tzw. Szybka Pamięć Taśmowa — program zmieniający system zapisu na szybszy. Dopuszcza różne prędkości, optymalna to 2—3 razy szybciej niż normalnie. Trzecia wersja TURBO to wbudowany w interfejs MASTERFACE moduł ze zmodyfikowanymi procedurami transmisji (ok. 2 razy szybciej). Wreszcie czwarta wersja to TURBO COMPRESS COPY autorstwa anonimowego Kato, skracający czas wczytywania o połowę i nie wymagający procedury ładującej. Czasem używane są też systemy polegające na przeróbce magnetofonu, współpracujące z jedną z podanych wcześniej wersji TURBO.







SERWIS KOMPUTERÓW

TEST

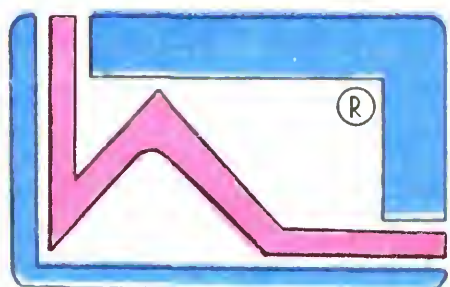
40-164 Katowice ul. Modrzewiowa 24/33

poleca naprawy:

- ATARI 600, 800, 65, 130 XL,XE
- COMMODORE 16, 116, +4, 64, 128
- DISC DRIVE 1541, 1570, 1571. 1050
- MAGNETOFONY COMMODORE
- DRUKARKI

godz. 9-11, 16-18

(SB 30)



WIENCEK ELEKTRONIK

PROFESJONALNY SERWIS KOMPUTERÓW

Katowice-os.W.Witosa ul. Ossowskiego 28 tel. 549-779

Poleca usługi w zakresie naprawy komputerów:
IBM PC XT, AT, 386,
COMMODORE plus 4, 16, 64, 116, 128, AMIGA, COLT,
ATARI 600, 800, 65, 130, 520, SPECTRUM,
magnetofony, sterowniki, drukarki, monitory, zasilacze.
Skupuje elementy elektroniczne i sprzęt.
ZAPRASZAMY OD 8.00—16.00 PONIEDZIAŁKI 10.00—18.00

K-269



- gry i programy użytkowe
- co piąty program gratis
- inne bonifikaty
- opisy gier i instrukcje
- pomoc dla początkujących
- wysyłka na cały kraj
- katalog gratis

WYPOŻYCZALNIA TOMBAT XL ATARI XE

Nasz adres:
ul. Magistracka 27 m 26
01-413 Warszawa
Tel. 363-078 godz. 12-20
Zapraszamy!

SB-12

Agencyjny Zakład Usługowy SPHW poleca usługi w zakresie:
- serwis komputerów SPECTRUM, C-64, TIMEX, XT, AT, rozszerzanie pamięci
- przestrajanie PAL-SECAM
- wejścia monitorowe TV

Warszawa ul. Puławska 102
tel. 44-87-89
czynny 12.00—19.00
Rachunki, gwarancja
ZAPRASZAMY

D-114

Naprawa komputerów ATARI, COMMODORE, IBM, SPECTRUM oraz urządzeń peryferyjnych. Warszawa tel. 22-07-85

D-84

ZX SPECTRUM, TIMEX, ATARI
- **programy użytkowe, edukacyjne, gry**
- **programy dla rzemiosła (receptury, kalkulacje, remanenty)**
- **instrukcje do programów**
- **informacje po nadesłaniu koperty zwrotnej ze znaczkiem**
- **wysyłka na cały kraj — rachunki**

„P.K.T.S." Studio Komputerowe
00-103 Warszawa
ul. Królewska 43 m 25

(SB-59)

MÓZGOPROCESOR! — rewelacyjną polską grę przygodową dla ZX Spectrum (program + kaseta + opis 3100 zł) otrzymasz pisząc: COMPUTER ADVENTURE STUDIO, Bochnia 32-700, ul. Kazimierza Wielkiego 37/45.

(SB 20)

—COMPUTER SERVICE

MS elektronik
naprawy komputerów:
Spectrum 48k, +, 128, +2, +3
Amstrad- Schneider
Sharp
Drukarki, Interfejsy
Wyjścia monitorowe
Czynne: od 9.00—16.00
MSelektronik Legionowa 23, 00-343 Warszawa
Dojazd: 105, 305, F (jelonki)
tel 37-76-65.

(K-118)

ATARI TURBO 2000 F

Nowy system transmisji danych z magnetofonem przyspieszony do 6700 bodów

Komplet:
- cartridge
- kaseta z 5 programami kopiującymi
- przeróbka magnetofonu
- 12 m-cy gwarancji za 29 000 zł oraz interfejs do standardowego magnetofonu za 22 000 zł

oferuje firma MUEL ul. Cząstkowska 30, 01-678 Warszawa tel. 33-40-91

D-121

ATARI 800 XL, 65 XE, 130 XE

Sprzedaż wysyłkowa
gier i programów użytkowych
na kasetach i dyskietkach.
Również w systemie TURBO 2000
Wszystkie nowości!!!
Instrukcje i literatura.
Dla zainteresowanych rachunki.

ANWIKOL

03-721 Warszawa ul. Jagiellońska 3/28.

(SB 74)

KOMPUTEROWE TESTY PRZYGOTOWUJĄCE DO EGZAMINÓW WSTĘPNYCH NA MEDYCYNĘ DLA ZX SPECTRUM, TIMEX, JUNIOR
POLECA „KOMPRED" Witkiewicza 15/12 59-220 LEGNICA

D-180/2

ZX Spectrum
SOUND-trójkanałowy, stereofoniczny, przelotowy interfejs muzyczny
KEMPSTON, SINCLAIR
Błyskawiczna wysyłka pocztą.
Ekspresowe naprawy klawiatur.
NiKUE ul. Meissnera 14 m 1
03-982 Warszawa tel. 15-93-38 w godzinach 18-20

D-191

ATASERW

43-100 TYCHY ul. Lencewicza 46/3
tel. 27 69 66

rozwiązania sprzętowe

do ATARI XL/XE:

1. TURBO DOS—wspaniały DOS na kartridżu
2. TOP DRIVE—do stacji 1050, LDW 2000, CALIFORNIA samodzielny montaż—wysyłkowo (rec. INFORMIK III/88)
3. INTERFEJS CENTRONIKS
4. ROZSZERZENIA PAMIĘCI
5. BASIC XE—kartridż
6. TURBO DOS + BUG65—kartridż

12 miesięcy gwarancji. Informacje i zamówienia telefonicznie (wtorek 8—12, środa, czwartek 16—18) i listownie po otrzymaniu koperty zwrotnej.

K—236



JOYSTICKI do Atari Commodore, Spectrum, Amstrad precyzyjny mechanizm specjalne styki 9800,– zł, 6 m-cy gwarancja interface do Spectrum

Wysyłka natychmiastowa za zaliczeniem pocztowym

Dla instytucji rachunki płatne przelewem

ELEKTROMECHANIKA

ul. Cegielniania 17
32-410 DOBCZYCE

G-129

ATARI ST, AMIGA
Tylko najlepsze programy
INSTRUKCJE!!!
własne unikalne opracowania.
Największy wybór w Polsce.
Fantazy, Adventure Club-światowe hity
porady, wskazówki, opisy, wymiana doświadczeń.
„KOMPUTER STUDIO" 04-202 Warszawa, ul. Marsa 6 tel. 15-42-20.

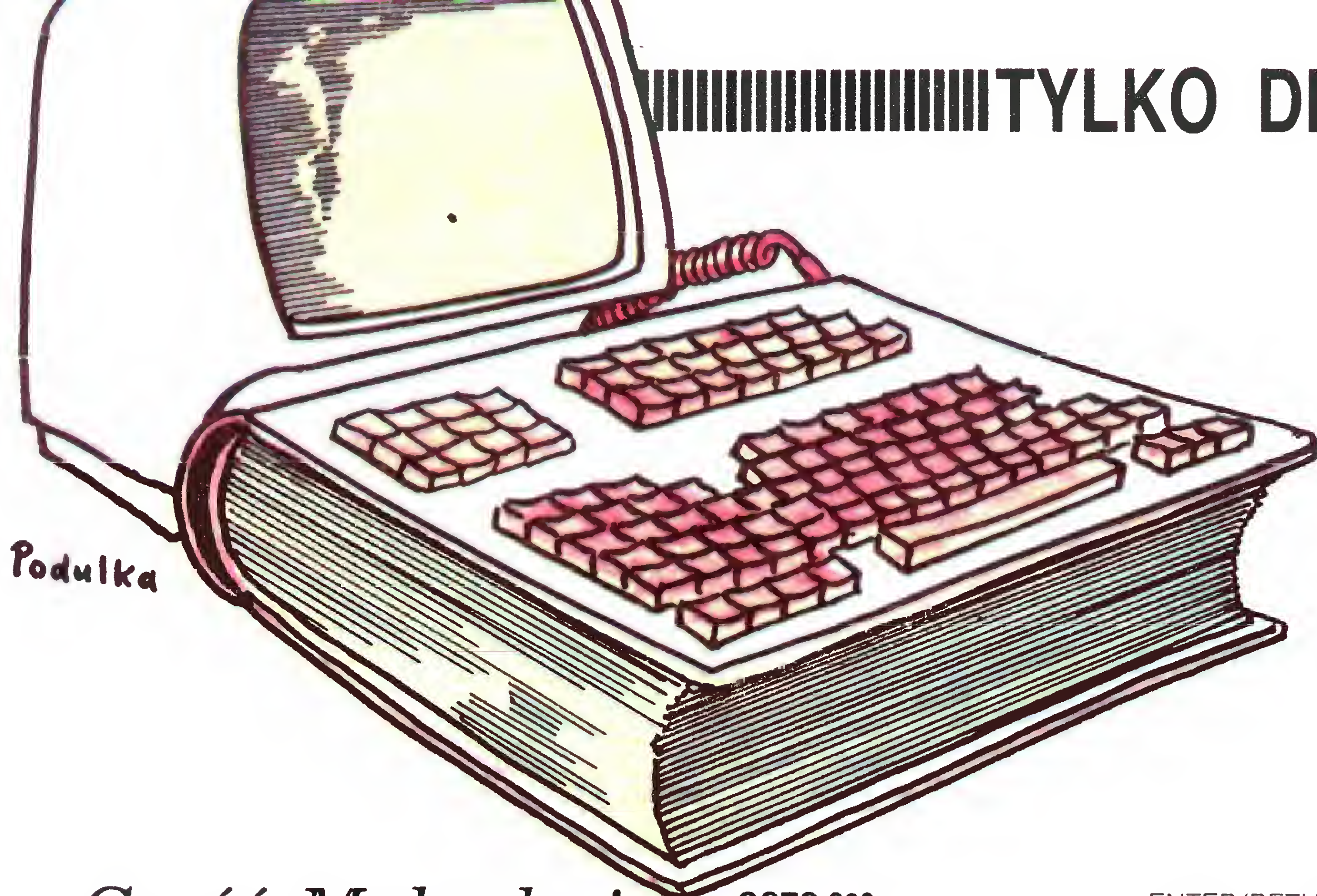
D-201

Atari XL/XE najnowsze programy, instrukcje oraz literaturę wysyła studio Mikrobit. Okazja, szybko i tanio. Marek Kąca, Malborska 6/160, 03-286 Warszawa.

(SB 75)

Atari XL/XE: Napiszę każdy program na zamówienie (Basic, Assembler), odbezpieczę dyski, programy kasetowe. Przemysław Kucharzewski, Plac Obrońców Pokoju 14/8, 66-200 Świebodzin, tel. 24382.

(SB 72)



ZAPISYWANIE I ODCZYTYWANIE

Cześć Maluchy!

W programach, które pisaliśmy dotychczas rozkazy zapisywanie (SAVE) i odczytywania (LOAD) stosowaliśmy jedynie w trybie bezpośrednim. Można jednakże stosować je również wewnątrz programów. Przynosi to często bardzo interesujące efekty.

Na początek zbudujemy komputerowy notes z adresami przyjaciół. Będzie się on sam zapisywał.

```
10 PRINT "Czytanie notesu .........GOTO 100"
20 PRINT "Dopisanie adresu ......... GOTO 200"
30 PRINT "Zapis na magnetofon ... GOTO 300"
40 GOTO 500
100 PRINT 1; "Zuzanka", "Czekoladowa 7"
110 PRINT 2; "Magdalena", "Ananasa 1 m. 9"
120 PRINT 3; "Kuba", "Karmelkowska 2/6"
130 PRINT 4; "Romek", "Al. Budyniu 13"
140 PRINT 5; "Jurek Z.", "Pod Gruszkami 5"

199 GOTO 10
200 LIST 100-190
300 SAVE "notes"
310 PRINT "Koniec zapisywania"
320 GOTO 10
500 :
```

Po uruchomieniu programu ukaże się następujący zestaw możliwości:

```
RUN                                   ›ENTER/RETURN‹
Czytanie notesu ...... GOTO 100
Dopisanie adresu ....... GOTO 200
Zapis na magnetofon ... GOTO 300
```

Jeśli chcemy wybrać którąś z funkcji, wpisujemy rozkaz **GOTO** z odpowiednim numerem linii. Wybierzmy na przykład „czytanie notesu".

```
GOTO 100                              ›ENTER/RETURN‹
1 Zuzanka Czekoladowa 7
2 Magdalena Ananasa 1 m. 9
3 Kuba Karmelkowska 2/6
4 Romek Al. Budyniu 13
5 Jurek Z. Pod Gruszkami 5
Czytanie notesu ...... GOTO 100
Dopisanie adresu ...... GOTO 200
Zapis na magnetofon ...... GOTO 300
```

Zestaw poleceń, czyli tak zwane menu (czytaj meni) drukowane jest ponownie. Po przeczytaniu odpowiedniej informacji z notesu możemy więc bez problemu wybrać następną funkcję — „dopisanie adresu".

```
GOTO 200                              ›ENTER/RETURN‹
100 PRINT 1; "Zuzanka", "Czekoladowa 7"
110 PRINT 2; "Magdalena", "Ananasa 1 m. 9"
120 PRINT 3; "Kuba", "Karmelkowska 2/6"
130 PRINT 4; "Romek", "Al. Budyniu 13"
140 PRINT 5; "Jurek Z.", "Pod Gruszkami 5"
```

Ukazuje się nam fragment programu zawierający adresy. Możemy uzupełnić program:

```
150 PRINT 6; "Wojtek", "Kwiatkowa 135"
                                      ›ENTER/RETURN‹
```

Przeglądnijmy jszcze raz nasz notes.

```
GOTO 100                              ENTER/RETURN
1 Zuzanka Czekoladowa 7
2 Magdalena Ananasa 1 m. 9
3 Kuba Karmelkowska 2/6
4 Romek Al. Budyniu 13
5 Jurek Z. Pod Gruszkami 5
6 Wojtek Kwiatkowa 135
Czytanie notesu .... GOTO 100
Dopisanie adresu .... GOTO 200
Zapis na magnetofon .... GOTO 300
```

i uzupełniony zapiszmy na kasetę.

```
GOTO 300                              ›ENTER/RETURN‹
Press REC&PLAY on the tape than any key:
```

Sprawdźmy jeszcze, czy w magnetofonie znajduje się właściwa kaseta i wykonajmy polecenie. Po chwili na ekranie ukaże się komunikat:

```
Koniec zapisywania
Czytanie notesu ... GOTO 100
Dopisanie adresu ... GOTO 200
Zapis na magnetofon ... GOTO 300
```

Jeszcze bardziej interesujące efekty daje zastosowanie w programie rozkazu **LOAD**. Możemy na przykład napisać kilka programów, które będę się wczytywały kolejno. W tej samej kolejności powinny być zapisane na taśmie. Skorzystamy ze wspaniałych wierszy **Jana Brzechwy**.

```
5 REM *********** Program 1 ************************
10 PRINT "! -------------------------------------------- !"
20 PRINT "!      ZOO                                     !"
30 PRINT "!    Jana Brzechwy                             !"
40 PRINT "! -------------------------------------------- !"
50 PRINT "! -------------------------------------------- !"
60 PRINT "! -------------------------------------------- !"
70 PRINT
80 PRINT "Po ukazaniu się kursora"
90 PRINT "Wpisz RUN i naciśnij
        ›ENTER/RETURN‹

100 LOAD "wiersz1"
SAVE "ZOO"                            ›ENTER/RETURN‹
Press REC& PLAY on the tape than any key:
Saving "ZOO"
```

I drugi program:

```
5 REM *********** Program 2 ***********
10 PRINT
20 PRINT "Małpy skaczą niedościgle,"
30 PRINT "Małpy robią małpie figle,"
40 PRINT "Niech pan spojrzy na pawiana"
50 PRINT "Co za małpa, proszę pana."
60 PRINT
70 PRINT "Po ukazaniu się kursora"
80 PRINT "Wpisz RUN i naciśnij
        ›ENTER/RETURN‹
90 LOAD "wiersz3"
SAVE "wiersz2"                        ›ENTER/RETURN‹
Press REC&PLAY on the tape than any key:
Saving „wiersz2"
```

A teraz trzeci:

```
5 REM *********** Program 3 ***********
10 PRINT
20 PRINT "Rudy ojciec, rudy dziadek,"
30 PRINT "Rudy ogon to mój spadek,"
40 PRINT "A ja jestem rudy lis."
50 PRINT "Ruszaj stąd, bo będę gryzł."
60 PRINT
70 PRINT "Po ukazaniu się kursora"
80 PRINT "Wpisz RUN i naciśnij
        ›ENTER/RETURN‹"
90 LOAD "wiersz3"
SAVE "wiersz2"                        ›ENTER/RETURN‹
Press REC&PLAY on the tape than any key:
Saving "wiersz2"
```

W ten sposób możemy zapisać całe ZOO Jana Brzechwy. Następnie cofnijmy taśmę, wczytajmy i uruchommy pierwszy program.

```
LOAD "ZOO"                            ›ENTER/RETURN‹
Press Play on the tape than any key:
Loading "ZOO"
RUN                                   ›ENTER/RETURN‹
!--------------------------------------------------------------!
!          ZOO                                                 !
!          Jana Brzechwy                                       !
!--------------------------------------------------------------!
!                                                              !
!                                                              !

Po ukazaniu się kursora
Wpisz RUN i naciśnij ›ENTER/RETURN‹
Press PLAY on the tape than any key:
Loading "Wiersz1"
RUN                                   ›ENTER/RETURN‹
Małpy skaczą niedościgle,
Małpy robią małpie figle,
Niech pan spojrzy na pawiana
Co za małpa, proszę pana.

Po ukazaniu się kursora
Wpisz RUN i naciśnij ›ENTER/RETURN‹
Press PLAY on the tape than any key:
Loading "Wiersz2"
RUN                                   ›ENTER/RETURN‹

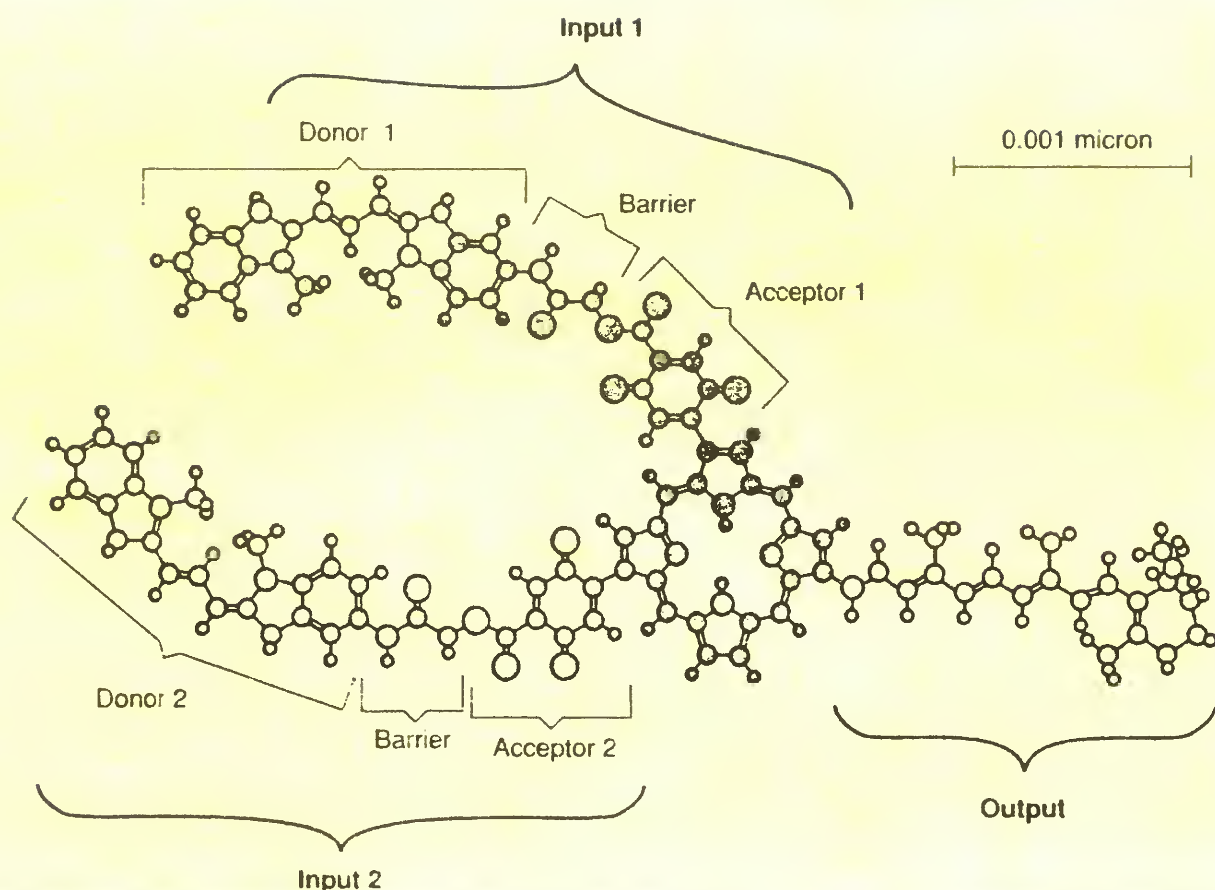
Rudy ojciec, rudy dziadek,
Rudy ogon to mój spadek,
A ja jestem rudy lis.
Ruszaj stąd, bo będę gryzł.

Po ukazaniu się kursora
Wpisz RUN i naciśnij ›ENTER/RETURN‹
Press PLAY on the tape than any key:
Loading "Wiersz 3"
```

I tak dalej...

Przyjemnej lektury życzy Wam

Romek



Rys. 1. Molekularna bramka typu NAND sterowana światłem laserowym. Ma ona rozmiar rzędu 4 nm i jest 10000 razy szybsza od jej krzemowego odpowiednika. Składa się z dwóch wejść (Input 1, Input 2) i jednego wyjścia (Output). Łańcuch odpowiadający każdemu z wejść zawiera trzy części: obszar donora, barierę i obszar akceptora.

MOLEKULARNY KOMPUTER

Już parę lat temu stwierdzono, że gdyby postęp w dziedzinie motoryzacji dorównywał postępowi w elektronice, to najnowszy model Rolls-Royce'a kosztowałby trzy dolary i na trasie Ziemia—Księżyc spalałby od 4 do 5 litrów paliwa.

I rzeczywiście, pierwsze komputery budowane na lampach elektronowych zajmowały dużej wielkości pomieszczenie, do ich zasilania potrzebna była elektrownia, a ich szybkość i pamięć operacyjna były mniej niż skromne.

Wprowadzenie technologii półprzewodnikowej i wynalazek układu scalonego zaowocowały po czterech dekadach burzliwego rozwoju urządzeniami, których sercem jest płytka krzemu o rozmiarach paznokcia. Komputery stały się bardzo szybkie, niezawodne, powszechnie dostępne i tanie.

Obecnie produkowane układy scalone bardzo dużej skali integracji (VLSI — Very Large Scale Integration) umożliwiają upakowanie 50 tys. tranzystorów w 1 mm^2. Odstępy między poszczególnymi elementami wynoszą 1—2 mikrony (10^{-6} m). Osiągane czasy przełączania, określające szybkość pracy, są rzędu kilkunastu nanosekund (10^{-9} s). Oczekuje się, że ulepszone technologie wytwarzania i nowe materiały pozwolą na zwiększenie gęstości upakowania do 0.1 mikrona.

Jeśli w dążeniu do dalszej miniaturyzacji schodzimy poniżej tej granicy, to wkraczamy w obszar rządzony mechaniką kwantową. Elektryczność, znana nam ze szkoły, jako kolektywny fenomen, przestaje być wystarczającym opisem zjawisk na tym poziomie — musimy uwzględnić efekty kwantowe dotyczące pojedynczych elektronów.

MOLEKULARNA BRAMKA TYPU NAND

Zejście do skali submikronowej, czyli do rozmiarów rzędu nanometrów (10^{-9} m), sprowadza się praktycznie do manipulowania cząsteczkami związków chemicznych. Obecność elektronów, wędrujących między poszczególnymi atomami molekuły zmienia istotnie jej fundamentalne własności fizyczne, w szczególności przewodnictwo elektryczne lub pochłanianie światła o określonej długości fali.

Zjawiska te wykorzystywane są przy tworzeniu prototypowych urządzeń elektroniki molekularnej (MED — Molecular Electronic Devices). Na rysunku 1 przedstawiono działający model bramki logicznej typu NAND skonstruowanej w oparciu o cząsteczkę światłoczułego pigmentu występującego w siatkówce oka ludzkiego. Molekuła tego związku chemicznego ma rozmiar 4 nm i kształtem swym przypomina małe widełki o dwóch odnogach i jednej szyjce. Każda z odnóg, składająca się z trzech części (obszar donora, bariera i obszar akceptora), jest wejściem bramki. Wyjściem tego układu jest szyjka.

Impulsy światła laserowego, o określonej długości fali zmieniają własności cząsteczki. Do właściwego działania modelu konieczne jest zastosowanie trzech laserów. Dwa z nich oświetlają wejścia bramki, natomiast trzeci laser oświetla wyjście.

W stanie podstawowym, gdy żadne z wejść nie jest oświetlone (logiczne zera), swobodne elektrony każdego z donorów znajdują się w obszarze akceptorów w pobliżu wyjścia bramki. Światło przechodzące przez tę część molekuły nie ulega absorbcji (logiczna jedynka na wyjściu). Oświetlenie obu odnóg wejściowych (logiczne jedynki) powoduje, że oba elektrony wędrują w obszar donorów zmieniając własności absorbcyjne szyjki wyjściowej. W tym niestabilnym stanie przejściowym wyjście pochłania światło laserowe o określonej długości. Sytuacja ta odpowiada logicznemu zeru na wyjściu bramki.

Na rysunku 2 przedstawiono sprzężenie prototypowego urządzenia elektroniki molekularnej ze światem zewnętrznym. Urządzenie to, posiadające 4 wejścia i 4 wyjścia, jest oświetlane ośmioma laserami o różnych długościach fali. Przechodzace światło, po dekompozycji w pryzmacie, jest analizowane przez rząd światłoczułych diod, z których każda reaguje na inny kolor (długość fali). Ponieważ padający promień światła ma większy rozmiar niż same cząsteczki, lokalizacja właściwej bramki i sygnału do niej przyłożonego jest określona nie przestrzennie, ale przez odrębne własności absorbcyjne różnych fragmentów molekuł.

INŻYNIERIA GENETYCZNA

Każda funkcja logiczna, np. dodawanie dwóch liczb, może być zrealizowana jako kombinacja wielu bramek typu NAND. Bawiąc się klockami LEGO możemy skonstruować dowolną budowlę posługując się kilkoma rodzajami podstawowych elementów. Podobnie projektowany jest procesor komputera, tylko że w tym przypadku jedynym „klockiem" może być właśnie bramka NAND.

Z przedstawionego opisu widać, że fundamentalnym problemem przy konstrukcji komputera molekularnego jest połączenie poszczególnych bramek ze sobą. Zagadnienie to nie zostało rozwiązane dotychczas w sposób zadawalający. Trudno wyobrazić sobie, że każdy element będziemy łączyć poprzez kilka laserów i diod światłoczułych.

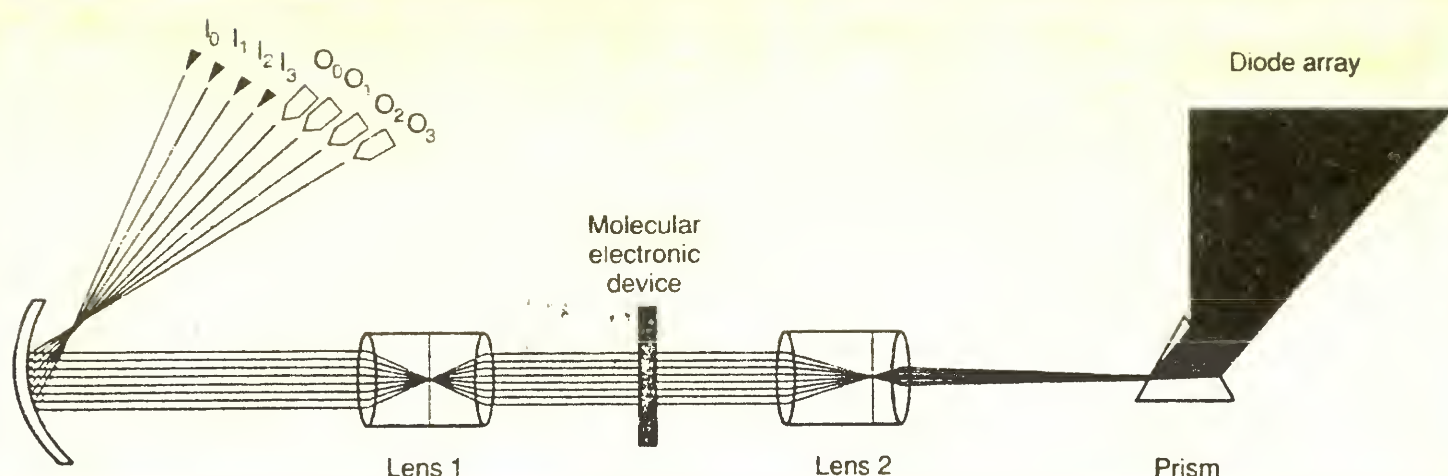
Z pomocą może nam przyjść inżynieria genetyczna. Genetycy są ludźmi, którzy od wielu lat budują nowe cząsteczki chemiczne, manipulując pojedynczymi atomami. Bardzo ciekawie wygląda propozycja wykorzystania do produkcji układów komputera molekularnego mechanizmu realizowanego przez rybosomy odczytujące informację genetyczną, na podstawie której komórki syntetyzują niezbędne białka. Jeśli pomysł ten zostanie urzeczywistniony, to potrzebne elementy komputera będą same powstawały w probówce zawierającej odpowiedni roztwór.

PERSPEKTYWY

Zbudowanie komputera molekularnego jest jeszcze kwestią czasu. Większość prowadzonych prac to badania podstawowe, którymi zajmują się czołowe laboratoria naukowe na świecie. Musimy poczekać 10—15 lat nim na rynku pojawią się handlowe produkty tej technologii. Komputery tej generacji będą 10 tysięcy razy szybsze od najszybszego obecnie komputera typu Cray, a zastosowane układy będą charakteryzowały się gęstością upakowania o 5 rzędów lepszą w stosunku do współczesnych elementów półprzewodnikowych. Złożone symulacje modeli fizycznych, rozpoznawanie mowy i obrazów w czasie rzeczywistym, przetwarzanie grafiki i rozumienie języków naturalnych, to te dziedziny, które najbardziej skorzystają z zalet komputera molekularnego.

Oprac. na podst. art. M.A. Clarksona, „Byte" 5,89

(JM)



Rys. 2 Sprzężenie molekularnego urządzenia elektronicznego (MED — ang. Molecular Electronic Device) ze światem zewnętrznym. Osiem częstotliwości światła laserowego (reprezentujących cztery wejścia i cztery wyjścia), przechodzących przez MED, jest odczytywane przez rząd diod światłoczułych (diode array). (Lens — soczewka, Prism — pryzmat).